HIGH-FIDELITY EMULATION OF SPATIOTEMPORALLY EVOLVING FLOW DYNAMICS

A Dissertation
Presented to
The Academic Faculty

by

Yu-Hung Chang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
Daniel Guggenheim School of Aerospace Engineering
College of Engineering

Georgia Institute of Technology
December 2018

HIGH-FIDELITY EMULATION OF SPATIOTEMPORALLY EVOLVING FLOW DYNAMICS

Approved by:

Dr. Vigor Yang, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. C. F. Jeff Wu
School of Information and System Engineering
Georgia Institute of Technology

Dr. Lakshmi N Sankar
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Roshan Vengazhiyil
School of Information and System Engineering
Georgia Institute of Technology

Dr. Joseph Oefelein
School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: September 6, 2018

DEDICATION

I would love to dedicate this work to my beloved parents for nursing me with affections and love and their dedicated partnership for success in my life. I also want to dedicate this work to my dear fiancé, Dr. Sarvepalli Devprakash Jokhai, who has never stopped supporting me and encouraging me during the darkest hours of my life.

ACKNOWLEDGEMENTS

After an intensive journal since last summer, today is the day: writing this note of appreciation is the finishing touch on my dissertation. It has been a period of intense learning for me, not only in the scientific arena but also on a personal level. Writing this dissertation has had a significant impact on me. I would like to reflect on the people who have unconditionally supported and helped me so much throughout this period.

I would first like to express my sincere gratitude to my advisor at Georgia Institute of Technology, Dr. Vigor Yang, for his continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. Dr. Yang, I want to thank you for guiding me through this wonderful journey for my doctoral program. Your excellent leadership and advisory guidance is the lighthouse while I was sailing in the darkness and couldn't find a way out of my research. I would like to thank for all of the opportunities I was given to conduct my research and further my dissertation at Georgia Tech.

In addition, I would like to thank my mentors, Dr. Liwei Zhang and Dr. Xingjian Wang, for your valuable guidance. You definitely provided me with the tools that I needed to choose the right direction and successfully complete my Ph.D. program and dissertation.

I would also like to thank my colleagues, Dr. Liwei Zhang, Dr. Xingjian Wang, Dr. Shiang-Ting Yeh, Dr. Simon Mak, Dr. Chih-Li Sung, and Yixing Li, for their wonderful collaboration. I would particularly like to single out Liwei and Yixing for your kindness and help in the research group. Liwei, thank you for being my mentor while I first moved

to Atlanta. Yixing, I am grateful for your kind help on emulation work when I have to work overseas.

I would also love to thank my parents and sisters, who are always there for me offering their wise counsel and sympathetic ears. I want to thank all my friends who have ever given me your encouragements during this exciting, struggling and intensive journey. Finally, my beloved and dearest fiancé, Dr. Sarvepalli Devprakash Jokhai. You are my biggest supporter while I suffered the darkest hours of my life. Thank you for being there for me. You are my best friend, best listener, and the best partner in my life. I feel dearly lucky and thankful that God let us meet with each other.

Thank you very much, everyone! I am beyond words in gratitude.

Yu-Hung Chang

Taipei, Taiwan June 24, 2018.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

A_n	cross-sectional area of the injection exit
A_{in}	the total inlet area
$\mathbb{C}$	CKSPOD covariance
K	geometric constant
L	Length
$\mathcal{M}$	The spatial eigenfunctions of CPOD (i.e., CPOD modes)
R	Radius
$\boldsymbol{R}$	Gaussian correlation function
T_∞	ambient temperature
T_{in}	inlet temperature
$\mathbf{T}$	CPOD covariance matrix
$\mathbf{T}^*$	KSPOD correlation matrix
$\boldsymbol{c_i}$	i^{th} design setting
$\boldsymbol{e_i}$	Unit vector
h	Liquid film thickness
$\dot{m}$	mass flow rate

p_∞ ambient pressure

t time

u Velocity

w Weighting function for kriging

α Spreading angle

β The temporal coefficients of POD

ϕ The spatial eigenfunctions of POD

$\mathcal{T}$ CKSPOD transfer matrix

SUMMARY

This dissertation utilizes a comprehensive interdisciplinary approach to demonstrate a paradigm for a novel design strategy for new generation engineering. Computational fluid dynamics (CFD), reduced-basis modeling, statistics, uncertainty quantification, and machine learning are employed to develop this strategy.

In the real world, designing a new product or device may require months or years. It is therefore crucial to develop more time-efficient strategies for reducing investigation and development costs. Using a rocket engine injector as an example, this dissertation addresses fundamental issues critical to the development of an efficient and robust capability for understanding, analyzing, and predicting fluid dynamics and enhancing the interpretation of physical characteristics for future propulsion systems. The presented work demonstrates recent breakthroughs in modeling and data analytics techniques to substantially improve modeling capabilities at many levels.

Due to the high-pressure requirements of cryogenic propellants, such as those of liquid rocket engines, physical experiments are expensive. Furthermore, it is difficult to observe the physical mechanisms of the combustion process via optical diagnostics. High-fidelity CFD, such as large eddy simulations (LES), has been employed for decades to better capture the flowfield and combustion characteristics that occur in rocket engines, but these computationally expensive calculations are impractical for design purposes. A 2D axisymmetric LES case, for instance, can take 6-14 days with 200-350 CPU cores in parallelization, which is extremely costly and time-inefficient. Further, a full-size 3D LES

case with the same grid resolution and CPU cores as a 2D case may take over a month to complete.

To develop an efficient design strategy for new generation engines, therefore, an interdisciplinary revolution, spanning fields from statistics to engineering, is needed. Taking a swirl injector as a demonstration example, Design of Experiment (DoE) is formulated based on few pivotal geometric design parameters and the corresponding ranges for each of these parameters. Drawing upon prior knowledge of the major contributing geometric parameters, the sample size is determined based on semi-empirical approaches, with a recommended six to ten simulations per design variable. This approach facilitates the design process and reduces the number of total sample points required to efficiently scrutinize the design space.

To effectively and efficiently examine the physical mechanisms and dynamic details of instantaneous flow features for a new swirl injector design, serial novel data reduction methods are developed and employed to reduce the data size while keeping dominating physics information. These methods include low-fidelity models such as common proper orthogonal decomposition (CPOD), kernel-smoothed proper orthogonal decomposition (KSPOD), and common kernel-smoothed proper orthogonal decomposition (CKSPOD). The reduced data are used to train the high-fidelity simulation models, and finally a kriging-based emulator is applied to predict the dynamics of the flowfield with various spatiotemporal characteristics, based on the new geometric design of an injector. These representative metamodeling techniques are found to be substantially improved the modeling capabilities at all levels. Recent breakthroughs in modeling and data analytics successfully capture turbulent dynamics in a swirl injector and yield

predictions more quickly than high-fidelity simulations. Most notably, CKSPOD, the latest proposed emulator, can achieve a turnaround time 34,000 times faster than LES in evaluating a new design point across 1,000 snapshots with only 10 CPU cores. Furthermore, the presented work conducts uncertainty quantification (UQ) theorems to examine the uncertainties (i.e., the accuracy and precision) of all models. Results of the UQ analysis reveal not only that the proposed models are qualitatively good comparing with simulation but also that they perform quantitatively well for spatiotemporal predictions.

The work described in this dissertation produces a suite of multi-fidelity modeling techniques for effective and efficient assessment of the dynamic behaviors of a practical system, with geometric details over a broad range of operating conditions. This approach can also be applied to other engineering systems involving complex turbulence dynamics, nano/micro fluid dynamics, combustion instabilities, manufactory industry, geological exploration, biomedical device invention, medicine, and other fields.

It is noted that this dissertation interpolates materials from three published or submitted papers [1-3] by Simon Mak, Chih-Li Sung, C. F. Jeff Wu, Xingjian Wang, Shiang-Ting Yeah, Vigor Yang, Liwei Zhang, and the author of this dissertation (note: all names are listed in alphabetical order by the surnames). Partial results for the CPOD-based emulation of the presented work have been published in *JASA* and *AIAA J.* in 2017 [1] and 2018 [2] respectively, with the author of this dissertation as a co-author who contributes most of the preliminary data physical mechanism exploration, data work organization, and partial coding works. Partial research results for the KSPOD in the presented work have been submitted to *J. Comp. Phys.* in early 2018 and are currently under review, with the

author of this dissertation as first author [3]. Meanwhile, partial results for the CKSPOD in the presented work are currently being prepared for submission to *J. Comp. Phys.* as a first-author paper.

Keywords: design study, high-fidelity simulation, kriging, data reduction, surrogate model, swirl injector

CHAPTER 1. INTRODUCTION

1.1 Overview

A timely design process with appropriate cost is always a primary goal when developing a new product in the engineering and industry field. To develop a new produce or to solve a problem in the real world, the research and development (R&D) process sometimes becomes the most expensive process for companies or government agencies. Based on a report by the 2016 EU Industrial R&D Investment Scoreboard released by the European Commission [4], R&D spending as a percentage of net sales is 2.8% for aerospace and defense, 5.9% for automobiles and parts, 8.4% for technology hardware and equipment, and 10.6% for computer software and services. Hence, it is important to develop a design strategy to reduce the cost of the design process.

The objective of this work is to develop an accurate and computationally efficient surrogate model (i.e., emulator) to predict datasets containing rich spatiotemporal physical mechanisms for effective design surveys in any design space. The study encompasses computational fluid dynamics (CFD), reduced-basis modeling, statistics, and machine learning. As a demonstration case, the flow evolution in a swirl injector with spatiotemporally evolving flow dynamics in a broad design space is presented.

For design assessment, physical experiments can be extremely expensive and time-consuming, especially for complex systems operating over a wide range of conditions. Moreover, it is hard to gain insight into underlying physicochemical mechanisms through measurements using currently available experimental techniques. To better capture flow

characteristics and identify design attributes, one solution is high-fidelity modeling and simulations such as large-eddy simulation (LES). The LES framework employed in the present work is capable of dealing with fluid flow and combustion dynamics over the entire range of thermodynamic states [5-9]. These simulations, however, are computationally expensive and impractical for use as a primary tool to survey the design space; an axisymmetric simulation of flow evolution in a simplex swirl injection with LES-grade resolution, for instance, may take about 100,000 CPU hours on the hexa-core AMD Opteron Processor 8431. The traditional trial-and-error based design practice is no longer practical. To enable the use of high-fidelity simulations for design evaluation, an effective model must be incorporated into the design process.

The first step toward the development of an emulation (surrogate) model is Design of Experiments (DoE), which achieves more realistic computation timelines for building a database. Here we consider a swirl injector as a demonstration example [10], as shown schematically in Figure 1. The detailed flow characteristics have been previously explored using LES techniques [11, 12]. DoE can be formulated based on several key geometric parameters and their respective ranges of consideration. This work focuses on the effects of these geometric parameters (that is, location and width of the tangential entry and injection angle) on the injector performance, as measured by the thickness and spreading angle of the liquid film at the exit of the injector [10-12]. The total sample size is determined using a $10d$ rule-of-thumb described by Loeppky et al. [13], which recommends ten simulations per design parameter where d means the total number of design parameters. This approach substantially reduces the number of total sample points required to survey the design space.

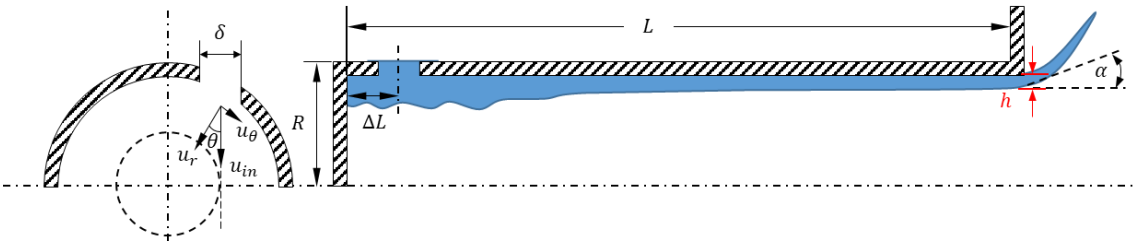

Figure 1—Schematic diagram for a simple swirl injector

The second step is the creation of a database with sufficient information to allow for a survey of the design space. This can be achieved by performing an LES-based high-fidelity simulation at the selected design points. For spatiotemporally evolving flows, however, the resulting database is too large to be handled effectively. Identification of dominant flow structures and reduction of the "big data" becomes essential for building the emulation model.

The final step requires the combination of POD and kriging methodologies. Kriging is a powerful machine-learning tool for interpolation and prediction [14]. The concept of kriging is to model unobserved responses using a Gaussian Process (GP) governed by a preset covariance function. The response surface of the training model can be evaluated via data-tuned weights to radial basis functions centered at observed points. In the present study, each simulation contains over 40,000 numerical grid points in the spatial domain. Kriging is required for each point if information for each point is to be modeled. To reduce the data size, proper orthogonal decomposition (POD) [15] has been incorporated into the prediction model (emulator).

Unfortunately, POD is only suitable for extracting instability structures at a single geometry, whereas for emulation, a method is needed that can extract common structures over varying geometries. To this end, three new decomposition algorithms — common proper orthogonal decomposition (CPOD) [1, 2], kernel-smoothed proper orthogonal decomposition (KSPOD) [3], and common kernel-smoothed proper orthogonal decomposition (CKSPOD) — are proposed in this work.

The key assumption of CPOD is that, under a physics-guided partition of the computational domain, the spatial distribution of coherent structures scales linearly over varying injector geometries. The CPOD-based emulation successfully predicts mean flow structures for swirl injectors with a broad range of geometric dimensions. To further predict spatiotemporal flow structures, KSPOD is developed to improve the prediction accuracy of flow evolution in the entire spatial domain and associated flow dynamics.

For KSPOD-based emulation, two main assumptions must be addressed. First, the physics extracted by POD modes in different cases are similar under the same rank, as determined by its energy. Second, the dominant modes capturing similar physics are transferred with same or similar phase, such that those physics would be retained through the kriging-weighted averaging based on the new design. KSPOD achieves three goals: (i) capturing turbulent flow dynamics; (ii) accurately predicting results as verified through quantitative comparisons with simulation results; (iii) yielding predictions with short turnaround times. The KSPOD-based emulator requires only about 0.02 CPU hour on an Intel Xeon Processor E5-1650 V4 to predict the flowfield at a new design point for the problem presented in the present study. The overall computation time, including data loading and training, is about 25 CPU hours (on an Intel Xeon Processor E5-1650 V4) to

predict the flow evolution at a new design point for a time duration of 10 ms with 1000 snapshots. For comparison, the corresponding LES calculation takes about 100,000 CPU hours (on the hexa-core AMD Opteron Processor 8431) for a spatial domain of over 40,000 numerical grid points.

It seems KSPOD conquer the problem to develop a time-efficient low fidelity model to capture spatiotemporal evolving flow characteristics for design surveys in a wide design space. However, KSPOD doesn't perform very well in some cases while this surrogate model fails on these two assumptions. Sometimes there's a discrepancy between the simulation and KSPOD-based emulation. The discrepancy is caused by time delay while phase differences are observed in POD modes between any two training cases under the same rank.

To fix the time-delay issues and improve the prediction accuracy of flow evolution in the entire spatial domain and associated flow dynamics, CKSPOD is developed here. This method takes the merits from both CPOD and KSPOD. The main reason that KSPOD-based emulation performs time-delay of evolving vortex structures along the injector is all training data deal with eigendecomposition at a different time [3]. In KSPOD algorithm, all training cases take eigendecomposition for POD individually before the kriging process. Although the individual POD process extract and preserves the most primary time and space information of evolving flows, this could create phase differences between any two POD modes under the same rank. The weighting number calculated by kriging can only partially fixed this problem. To solve this issue, CKSPOD applies the concept of CPOD to do eigendecomposition for all training cases once at the same time. Since all simulations (i.e., the training cases) share the same eigenvalues, no phase differences of POD modes

would appear between any two designs. Then, all cases apply the same eigendecomposition results to estimate their POD modes and coefficients individually for the following kriging process applied in KSPOD.

The CKSPOD-based emulator requires only about 0.035 CPU hour on an Intel Xeon Processor E5-1650 V4 to predict the flowfield at a new design point for the problem presented in the present study. The overall computation time, including data loading and training, is about 25 CPU hours (on an Intel Xeon Processor E5-1650 V4) to predict the flow evolution at a new design point for a time duration of 10 ms with 1000 snapshots. For comparison, the corresponding LES calculation takes about 100,000 CPU hours (on the AMD Opteron Hexa-Core Processor 8431) for a spatial domain of over 40,000 numerical grid points.

The work described in this dissertation adopts three sets of simulation runs that two sets for nonreacting flows and one set for reacting flows. These simulations are produced by a high-fidelity surrogate modeling technique for efficient prediction of complex flowfields over a broad range of operating conditions and geometric parameters. The dissertation is structured as follows. Chapter 1 introduces the history and related background for DoE and emulation. Chapter 2 states the theoretical framework including a modern design strategy for spatiotemporal evolving flow. Chapter 3 addresses the details of research design, including schematic configuration of the demonstrated injectors, DoE methods, three kriging-based surrogate models, and data-driven framework for emulation analysis. Chapter 4 shows research results and discussion for emulations. Chapter 5 concludes all works presented in this dissertation and future works.

1.2 The History of Design of Experiment (DoE)

The origin of design of experiment (DoE) can be traced back to the early 20th century, when the development of modern statistics was booming in the United Kingdom. During this time, statistics had been transformed into not only a useful tool for experimental and social sciences but also a rigorous mathematical discipline used for analysis in industry and politics. Quantitative evidence provides an excellent overall picture of a population or a geographical region, and qualitative evidence is often collected in small studies and based the experiences of a very few individuals. However, the later one can supply richer, deeper and broader information based on a few individuals or case examples. Hence, when the tools of statistical inference have helped to cultivate a new ideal of objectivity in scientific knowledge, qualitative evidence has gradually been replaced by quantitative results.

The keystone of DoE was laid in the UK in the first half of the 20th century [16] [17] by Sir Ronald Fisher, a British mathematician, statistician, and geneticist. From 1911–1933, Sir Fisher worked at Rothamsted Experimental Station (now called the Rothamsted Research), where agricultural research was reshaped by Fisher's methods. The improvement of crop production, analysis of variance, and experimental design required new practices and instruments in field and laboratory research and imposed a redistribution of expertise among statisticians, experimental scientists, and the farmers.

Eventually, the application of statistical methods in agriculture, as implemented at Rothamsted Experimental Station, made computing an integral activity to experimental research and permanently integrated the statistics tools and expertise into the station's research program. Fisher's statistical methods were not confined within agricultural studies. By the mid-20th century, his methods had become permanently ingrained in fields

including psychology, sociology, medicine, engineering, economics, quality control, education, chemistry, to name a few.

Immediately after World War II, the agriculture origins of DoE ended, and the first industrial era marked another renaissance in the use of DoE. In 1951, George E.P. Box and K. B. Wilson coauthored a paper [18] on response surface methodology which proposed the output as a response function to find the optimum conditions for a process via a sequence of designed experiments. Although Box and Wilson acknowledged that this model can produce only an approximation, they nonetheless used it because the model is simple to estimate and apply, even when little is known about the process. This paper represents a turning point that determined the shape of modern DoE.

In the 1950s, the importance of statistical quality control was taken to Japan by W Edward Deming [19]. After World War II, many viewed Japanese products negatively. To alter this impression that Japanese products were cheaply made and of poor quality, a series of statistical quality control methodologies was adopted by the Japanese industrial market.

In the 1960s, the quality of Japanese products began to improve significantly. The Japanese car industry implemented statistical quality control procedures and conducted experiments, which ushered in a new era. Total quality management [20] and continuous quality improvement are management techniques that have emerged from this statistical quality revolution, which combined statistical quality control and DoE. Thus, the Second Industrial Era (late 1970s–1990) of DoE began, also known as the quality revolution.

Genichi Taguchi (January 1, 1924–June 2, 2012), a Japanese engineer and statistician, discovered and published many of the techniques that were later brought to the

West. One of his famous statistical inventions, the Taguchi Orthogonal Arrays Design, is similar to the fractional-factorial designs in the West [21, 22]. This is a highly fractional orthogonal design based on a design matrix proposed by Taguchi that allows for the consideration of a selected subset of combinations of multiple factors at multiple levels. Taguchi introduced a concept of robust parameter design and process robustness. The concept is, nowadays, the well-known Taguchi methods applied to improve the quality of manufacturing processes and, more recently, has also been adopted by biotechnology, engineering, marketing, and advertising. His work includes three principal contributions to statistics: 1) a specific loss function; 2) the philosophy of off-line quality control; and 3) innovations in DoE. Although the Taguchi method has been controversial among some Western statisticians, most have accepted his proposed concepts, and his methods have been broadly implanted into Western industrial fields.

From around the late 1980s and early 1990s, Taguchi's and other statistical methods were no longer sufficient for manufacturing. Six Sigma, a set of techniques and tools for process improvement (i.e., a new method of representing continuous quality improvement), quickly rose to prominence. Six Sigma was first employed by engineer Bill Smith while working for Motorola in 1986 [23]. Later, this method was highly recognized and centralized within the business strategy at General Electric by their 8th CEO, John Welch, in 1995 [23].

The term "Six Sigma" originates from terminology associated with the statistical modeling of manufacturing processes. The maturity of a manufacturing process can be described by a sigma-rating indicating its yield or the percentage of defect-free products it creates. A Six Sigma process is one in which 99.99966% of all opportunities to produce

some feature of a part are statistically expected to be free of defects (i.e., 3.4 defective features per million opportunities). It incorporates many previous statistical and management techniques. Today, many businesses use this technique to apply statistics to make decisions based on quality and feedback loops.

A schematic diagram of the four eras in the history of DoE is plotted in Figure 2. Overall, DoE is a design mission to undertake experiments with the most efficient resources to obtain the most analytical and effective results. It aims to predict outcomes by introducing a change of the preconditions (i.e., controllable conditions) operated by one or more independent variables (i.e., input variables, or predictor variables). The change of one or more independent variables is generally hypothesized to result in a change in one or more dependent variables, also referred to as "output variables" or "response variables." This branch of applied statistics carries out the planning, conducting, analyzing, and interpreting of controlled tests to evaluate the factors that control the values of one or multiple parameters through a quick estimation.

In this work, the sample device is a simple injector with spatiotemporal evolving flow. By applying DoE, a well-designed data vault with simulation cases based on a range of design space across the physics phenomena of parameters of interests is built. Here, the parameters of interests within the design space are the "input variables" or "predictor variables;" the physics phenomena that can be quantified are the "output variables" or "response variables." After DoE, a data vault, which is the collection of data from all designed experiments, is utilized to build an emulator to predict the physics phenomena relevant to any new parameters within the design space.

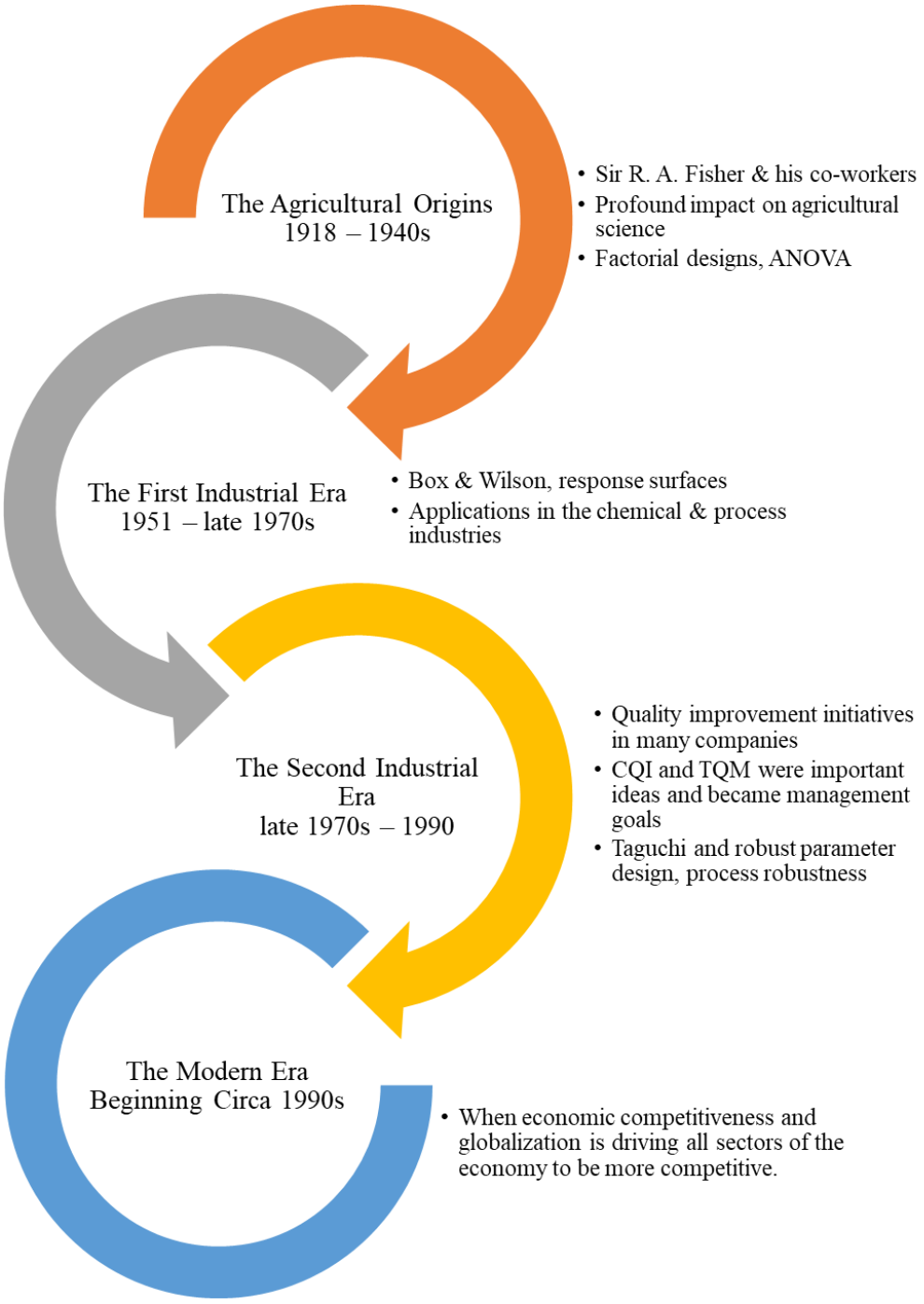

Figure 2—Schematic diagram of the four eras in the history of DoE

1.3 A Quick Overview of DoE Methods

Determining DoE revolves around the understanding of the effects of different variables on other variable(s). In mathematical terms, the objective is to establish a cause-and-effect relationship between several independent variables and a dependent variable of interest. In the context of DoE, the dependent variable is called an observation or response, and the independent variables are called design parameters or factors. Experiments are operated at different factor values, called levels. Each run of an experiment involves a combination of the levels of the factors under investigation. Each of the combinations is referred to as a treatment. In a single-factor experiment, each level of the factor is referred to as a treatment. In experiments with many factors, each combination of the levels of the factors is considered a treatment. When the same number of response observations are noted for each of the treatments of an experiment, the design of the experiment is said to be balanced. Repeated observations at a given treatment are referred to as replicates. The number of treatments of an experiment is determined based on the number of factor levels being investigated in the experiment. For example, if an experiment involving two factors is to be performed, with the first factor having x levels and the second factor having y levels, then xy treatment combinations can possibly be operated, and the experiment is an xy factorial design. If all xy combinations are undertaken, it a full-factorial experiment. However, if only some of the xy treatment combinations are run, the experiment is called a fractional factorial. In a full-factorial experiment, all factors and their interactions are under investigation, whereas in a fractional-factorial experiment, not all interactions are under consideration because not all treatment combinations are operated.

The size of an experiment can escalate rapidly when the number of factors or the number of the levels of the factors increases. For instance, if three factors at four levels each are to be used, 64 different treatments are required for a full-factorial experiment ($4 \times 4 \times 4 = 64$). If a fourth factor with five levels is added, 320 treatments are required ($4 \times 4 \times 4 \times 5 = 320$), and 640 treatments are required if a fifth factor with two levels is applied ($4 \times 4 \times 4 \times 5 \times 2 = 640$). If only two levels are used for each factor, then in the ten-factor case, 1,024 treatments are required ($2^5 = 2014$). For this reason, many experiments are restricted to two levels. Fractional-factorial experiments further reduce the number of treatments to be executed in an experiment. A 2^3 design with factors A, B, and C is plotted in Figure 3 as an example. This design tests three main effects ($k = 3$, k represents the number of total factors), A, B, and C; three two-factor interaction effects (i.e., $\binom{k}{2} = \binom{3}{2} = 3$), AB, BC, and AC; and one three factor interaction effect (i.e., $\binom{k}{1} = \binom{3}{1} = 1$), ABC. Therefore, the design requires eight runs per replicate. The eight treatment combinations corresponding to these runs are (1), a, b, ab, c, ac, bc, and abc. Note that the treatment combinations are written in such an order that factors are introduced one by one with each new factor being combined with the preceding terms. This order of writing the treatments is called Yates' order (also known as the "Standard Order"). The significance of Yates' order is that it facilitates the determination of the algebraic signs of the coefficients needed for calculating the main and interaction effects of each factor in a factorial experiment. The 2^3 design is shown in Figure 3(a) below and the design matrix is shown in Figure 3(b). The design matrix can be constructed by following the standard order for the treatment combinations to obtain the columns for the main effects and then multiplying the main effects columns to obtain the interaction columns.

(a)

Treatment Name	Factors		
	A	B	C
(1)	-1	-1	-1
a	1	-1	-1
b	-1	1	-1
ab	1	1	-1
c	-1	-1	1
ac	1	-1	1
bc	-1	1	1
abc	1	1	1

(b)

(1)	a	b	ab	c	ac	bc	abc
1	-1	-1	1	-1	1	1	-1
1	1	-1	-1	-1	-1	1	1
1	-1	1	-1	-1	1	1	1
1	1	1	1	-1	-1	-1	-1
1	-1	-1	1	1	1	-1	1
1	1	-1	-1	1	-1	1	-1
1	-1	1	-1	1	1	1	-1
1	1	1	1	1	-1	-1	1

Figure 3—A two-level factorial experiment design with three factors; (a) represents the 2^3 design and (b) represents the design matrix

The following quickly summarize some of the most common DoE categories:

1.3.1 One-Factor Designs

As the name implies, with this design, only one factor is under investigation, and the objective is to determine whether the response is significantly different at different factor levels. The factor can be either qualitative or quantitative. For qualitative factors (e.g., different suppliers, different materials, different labors, etc.), no extrapolations (i.e., predictions) can be operated outside the testing levels, and only the effect of the factor on the response can be determined. For quantitative factors, data from tests (e.g., weight, temperature, voltage, pressure, etc.) can be employed for both effect investigation and prediction, provided that sufficient data are available.

1.3.2 Factorial Designs

In factorial designs, multiple factors are under investigation simultaneously during the experiment. Moreover, qualitative factors, quantitative factors, or both can be considered. The objective of factorial designs is not only to identify the factors that significantly influence the response but also to investigate the effect of interactions between factors. Predictions can also be performed when quantitative factors are present, but this must be done carefully because certain designs limit the choice of predictive model. For instance, in two-level designs, only the linear relationship between the response and the factors can be used, which may not be pragmatic.

1.3.2.1 General Full-Factorial Designs

Generally, in full-factorial designs, each factor can have a different number of levels, and the factors can be either quantitative, qualitative, or both.

1.3.2.2 Two-Level Full-Factorial Designs

Two-level full-factorial designs restrict the number of levels for each factor to two. Compared to a general full-factorial experiment, the restriction to two levels reduces the number of treatments necessary to run a full-factorial experiment and allows for the investigation of all factors and all their interactions. If all factors are quantitative, the data from such experiments can be used for predictive purposes, provided a linear model is appropriate for modeling the response. It is noted that since only two levels are used, curvature cannot be modeled.

1.3.2.3 Two-Level Fractional-Factorial Designs

This type of design comprises a special category of two-level designs where not all factor level combinations are considered. In two-level fractional-factorial designs, the investigator can choose which combinations are to be excluded for certain interactions of interest.

1.3.2.4 Plackett-Burman Designs (1946)

A special category of two-level fractional-factorial designs was proposed by R. L. Plackett and J. P. Burman [24] while they were working in the British Ministry of Supply. Plackett-Burman designs are used for screening experiments because, in this design, main effects are, in general, heavily confounded with two-factor interactions. Hence, only a few specifically chosen experiment runs are performed to investigate only the main effects (i.e., no interactions).

1.3.2.5 Taguchi Orthogonal Arrays Design (1987)

Taguchi Orthogonal Arrays designs [25] are highly fractional and are applied to estimate main effects using only a few experimental runs. These designs are not only applicable to two-level factorial experiments but also can investigate main effects when factors have more than two levels. Designs are also applicable to investigate main effects for certain mixed level experiments when all the factors do not have the same number of levels.

1.3.3 Response Surface Method Designs (1951)

As mentioned in Section 1.2, the response surface methodology design, which is a collection of mathematical and statistical techniques for empirical model building, was introduced by George E. P. Box and K. B. Wilson in 1951 [18]. This special design method is used to determine the settings of the factors to achieve an optimum value of the response.

1.3.4 Reliability DoE

Reliability DoE (R-DoE) is a special category of DoE specifically intended to consider a response that is a life metric (e.g., age, miles, cycles, etc.) when the data may contain censored observations (suspensions, interval data). Only one response (typically failure time) is measured, but the designs can accommodate data sets that include suspensions (different censoring results), uncertainty as to when the units failed, or both in addition to complete data sets in which all the units under test failed and the failure time for each unit is known. Traditional DoE techniques usually assume that response values at any treatment level are normally distributed; on the contrary, R-DoE generally applies Weibull, lognormal, or exponential distributions to analyze data.

1.3.5 DoE for Computer Experiments

In the 21st century, the improvement of hardware allows increasing numbers of scientific phenomena to be studied by complex computer models (i.e., codes). In many cases, the basis of a computer model is a mathematical theory that implicitly relates the response to the factors. With the integration of appropriate computer hardware and software, it becomes possible to solve a mathematical system accurately with suitable

numerical methods. A computer experiment represents several runs of a code with various inputs and different outputs. One problem of computer experiment is, in many cases, that the codes are computationally expensive to run to fit a predictor of the output to the data. The objective of DoE application for a computer experiment is to provide a statistical basis for designing experiments (choosing the inputs) for efficient prediction with minimum runs. With this approach, estimates of uncertainty of predictions are also available.

In this work, the design and analysis of computer experiments are applied. A swirl injector is adopted as an example and the experiments are collected by simulation. Hence, a DoE method for computer experiments is needed. Further details are elaborated in the following section.

1.4 Kriging (Gaussian Process Based Model)

Kriging, a GP-based model, is a commonly-used method of interpolation for spatial data problems and has been applied with great success in a variety of fields [26]. The data are a set of observations of some variable(s) of interest, with some spatial correlation present. Usually, the results of kriging are the expected value ("kriging mean") and variance ("kriging variance") computed for every point within a region. In practice, to get better interpolation results (i.e., prediction), this must be done on a sufficiently fine grid. This method is named after Daniel G. Krige (August 26, 1919–March 3, 2013), who pioneered the field of geostatistics and was a professor at the University of the Witwatersrand, Republic of South Africa. Early in his career, Daniel Krige developed empirical statistical methods to predict ore grades from spatially correlated sample data in

the gold mines of South Africa [27, 28]. His approach was formalized by Georges Matheron in the 1960s (December 2, 1930–August 7, 2000) [29, 30]. Since then, pedologists, hydrologists, geologists, atmospheric scientists, and scientists and engineers in other fields have recognized the value of this technology in their own fields and contributed to making this technique more mature [31-37]. Today, kriging is applied widely and with increasing sophistication in public health, fishery, petroleum engineering, mining, geology, meteorology, hydrology, soil science, precision agriculture, pollution control, ecology, computer science, and many other fields. Kriging has become a generic term for several closely related least-squares methods that provide not only best linear unbiased predictions but also some nonlinear types of predictions. In the first half of the 20^{th} century, kriging rose to prominence and spurred major advancements of mathematical methods of interpolation across many fields.

Unlike other traditional interpolation methods in statistics, such as inverse distance weighted and Spline, to use the kriging effectively involves an interactive investigation of the spatial behavior of the phenomenon (prediction) represented by training data sets (observation) before selecting the best estimation method for generating the output surface. The inverse distance weighted and Spline interpolation methods are considered deterministic interpolation methods, in that they are directly based on the surrounding measured values or on specified mathematical formulas that determine the smoothness of the resulting surface. Conversely, kriging can build a statistical model that includes autocorrelation. A kriging model considers the statistical relationships among all the measured points; hence, a kriging model not only has the capability of producing a

prediction surface but also decreases the errors and provides better certainty or accuracy of the predictions.

The major assumption of a kriging method is that the distance or direction between sample points reflects a spatial correlation that can be used to explain variation in the surface. A kriging model fits a mathematical function to a specified number of points, or all points within a specified region, to determine the output value for the assigned location. Kriging is a multistep process; it includes the exploratory statistical analysis of the data, variogram modeling, creating the surface, and (optionally) exploring a variance surface. Kriging is the most applicable method when a spatially correlated distance or directional bias exists in the data. The nature of kriging allows it to be widely applied in fields like soil science, petroleum engineering, mining, and geology.

When kriging was first created for geostatistics, the model naturally considered only two- or three-dimensional inputs. However, with more application, a real-world problem can be much more complex than a case with two- or three-dimensional inputs. Today, the concept of Gaussian process regression (GPR) contains kriging and other GP-based mathematical tools derived from kriging. These tools can be applied for investigations involving not only multi-dimensional inputs but also time series.

These models are nonparametric kernel-based probabilistic models. Formally, a GP generates data located throughout some domain such that any finite subset of the range follows a multivariate Gaussian distribution. The most mathematically basic and the least general form of GPR can be considered as simple kriging, which assumes the expectation value is zero everywhere.

In the least general case, it considers a training set $\{(x_i, y_i); i = 1,2, \dots, n\}$, where $x_i \in \mathbb{R}$ and $y_i \in \mathbb{R}$, drawn from an unknown distribution. A GPR model addresses this question of predicting the value of a response variable $\boldsymbol{y}_{new}$ given the new input vector $\boldsymbol{x}_{new}$ and the training data. A linear regression model takes the form $\boldsymbol{y} = \boldsymbol{x}^T\beta + \epsilon$, where $\epsilon \sim N(0, \sigma^2)$. Here, $E[f(x_i)] = 0$ for simple kriging, as previously addressed. The error variance σ^2 and the coefficients β are estimated from the training data (observation). A GPR model explains the response by introducing latent variables, $f(x_i), i = 1,2, \dots, n$, from a GP and explicit basis functions, h. The covariance function of the latent variables captures the smoothness of the response and basis functions project the inputs $\boldsymbol{x}$ into a p-dimensional feature space.

For a more complex scenario, such as ordinary kriging, the model assumes a constant unknown mean only over the search neighborhood of x_0 (unknown), with $E[f(x_i)] = E[f(x_0)] = m$, where m is the unknown mean. More details about kriging and GP are found in Appendix I.

1.5 Uncertainty Quantification (UQ)

Statistical tools with data-driven analysis intended for computational simulation of complex real-world processes (e.g., supercritical combustion inside an engine, liquid fluid mixing of a small injector, etc.) are fundamental to virtually every field of science, engineering, medicine, and business. However, a statistical model of a process can rarely be established and implemented with assurance from its construction that the model accurately represents, emulates, or predicts the complete process. Thus, it is crucial to understand the uncertainties inherent in applying a model for predicting real-world

processes. Moreover, constructing a model usually requires extensive use of data concerning the real process being modeled (e.g., LES for supercritical combustion investigation). This interface of mathematical modeling, data, and uncertainty is known as UQ and has become a significant part of applied mathematics, engineering, and statistics.

As a modern interdisciplinary science, UQ has become very popular in the past decade. It passes through traditional research groups such as system uncertainty of measurements and incorporates statistics, numerical analysis, and computer science, applied mathematics, and all other applications in fields including biology, medicine, etc. Uncertainty regarding the model and its predictions accompanies every attempt to model complex real-world problems with mathematical and computer tools. This uncertainty sometimes arises from the numerical methods employed to approximate solutions to complex processes. In some cases, the processes are truly random and the uncertainty is inherent to the problems and cannot be reduced (e.g., turbulent flow). In other cases, the uncertainty stems from lacking knowledge about the underlying physics of the real-world problem. Moreover, the uncertainties can be derived by unknown, unmeasurable, or only indirectly observed inputs of a model.

Overall, the science of UQ is the end-to-end study of the influences of all forms of error and uncertainty in the models arising in many applications including statistical uncertainty for experiments of different runs and systematic uncertainty for the accuracy of some measurements. The development and analysis of UQ is useful not only for solving forward problems with uncertain inputs (i.e., propagating uncertainty in model inputs to model outputs) but also for solving inverse problems (i.e., where unknown model inputs are to be estimated from possibly noisy observations of model outputs). The considerations

of UQ research range from fundamental, mathematical, and statistical questions to practical questions of computational accuracy and cost [38]. In this dissertation, three different POD-based emulators are proposed and UQ analysis is applied to discuss the accuracy and precision of the prediction by each model.

CHAPTER 2. THEORETICAL FRAMEWORK

This chapter demonstrates the theoretical framework including a design strategy for modern engineering and the high-fidelity simulation theory by using a simple injector as example. Section 0 dissects all steps of the modern design strategy for procedures with spatiotemporal evolving flows; and section 2.2 depicts the high-fidelity simulation methodology applied for experimental data set collection.

2.1 A Modern Design Strategy for Spatiotemporal Evolving Flow

For design assessment, physical experiments can be extremely expensive and time consuming, especially for complex systems operating over a wide range of conditions. Therefore, for help in solving engineering design problems, a surrogate model is needed to evaluate design objectives and constraint functions as a function of design variables. One of the most popular surrogate models, CFD, has been applied for decades. One solution for better capturing flow characteristics and identifying design attributes is high-fidelity modeling and simulations such as large-eddy simulation (LES). The LES framework employed in the present work is capable of dealing with fluid flow and combustion dynamics over the entire range of thermodynamic states [5-9]. These simulations, however, are computationally expensive and impractical for use as a primary tool to survey the design space; an axisymmetric simulation of flow evolution in a simplex swirl injection with LES-grade resolution, for instance, may take about 100,000 CPU hours on the hexa-core AMD Opteron Processor 8431. As such, the traditional trial-and-error based design practice is no longer practical. To enable the use of high-fidelity simulations for design

evaluation, a new effective surrogate model, which is able to predict spatiotemporal evolving flow dynamics, must be incorporated into the design process [1, 2].

However, the current design strategy for most devices only provides emulators to estimate certain response analyses and assist in making decisions of optimal design. These emulators, such as Gaussian process (GP) based regression models, cannot play a role in prediction for spatiotemporal results, such as turbulent flowfields, to speed up the design-adequate decision process. Moreover, the final case must be generated by CFD, which still requires weeks or months.

To alter this situation and make the whole design process more time-efficient, the emulator with data-driven analysis must play a role in the design-adequate decision process and final case generation. Furthermore, it can save much time if highly accurate predictions from emulation can be used for further deepening case studies such as flow dynamic or instability analysis. Figure 4 shows a schematic diagram of a novel design strategy for a swirl injector using the aforementioned DoE and emulator. It can be dissected into four modules, with the following seven steps:

Module 1: Design of Experiment (DoE)

The first step is to identify the problem, including the design parameters and related response that are the essentials of designing a model. Then, the design space (i.e., the range of each design parameter) and the number of design parameters, d, must be determined. If the model contains complex physical information that may vary significantly regarding design space, a smaller range of design space is recommended. Based on the $10d$ rule of thumb [13], the optimal-minimum total

sample for training data is ten times the number of design parameters (i.e., $10d$). However, if the design space is broad, a total sample (i.e., design points) of more than $10d$ is suggested.

A design matrix for a specified design space is generated by an optimized DoE algorithm. All training data are produced by sample points from the design matrix. If the total number of training samples is fixed (e.g., $10d$), an optimized DoE methodology with the best uniform design performance (e.g., MaxPro [39]) is prescribed. All design points arranged from DoE should be applied for the experiment.

If the total number of training tests may change (i.e., $> 10d$), the Sliced Latin Hyper Cube Design (SLHD) methodology is recommended to generate m design points, where $m > 10d$. To efficiently build the sample data sets, applying the first $10d$ design points from DoE for experiment is suggested. The remaining $10d - m$ design points can be added to the sample data vaults until more training data sets are needed to improve model validation.

Module 2: Data Vault

All training cases are produced via experiment based on the uniform design matrix. In the case demonstrated in this dissertation, the experiment is numerical simulation. During this process, one or more cases can be made simultaneously for a later model validation step. It is also crucial to build several cases outside of the design matrix, but within the design space, for later validation.

Step 3: Emulation

1) Data Analysis

To successfully establish a surrogate model for emulation, it is important to analyze the training cases to first fully understand the physical characteristics of data vaults. Understanding the physics behind the training data set is helpful in selecting the data-reduction methodology.

2) Data Reduction

To efficiently build an emulator with an accurate prediction function, data reduction is necessary to later increase the efficiency of emulation process. Here, proper orthogonal decomposition (POD) is applied to extract dominant physics information and flow dynamic information, which can be applied to train a surrogate prediction model. However, POD is based on a second-order linear statistics algorithm, which is capable of extracting information from snapshots of the flow field and is thus applicable to experimental data [40, 41]. It is also useful to extract turbulent coherent structures [42].

However, drawbacks to POD also exist. For example, energy may not be the correct measure to rank the flow structures in all circumstances. In addition, due to the choice of second-order statistics as a basis for the decomposition, valuable phase information could be lost. To further improve the design strategy, a more improved decomposition methodology is required.

3) Surrogate Model Training

After decomposing raw training samples and extracting essential information, a kriging-based approach can be applied to train a surrogate model for

efficient emulation. In this work, the new concept of kernel-smoothed proper orthogonal decomposition (KSPOD) is used to compile all important unstable features from all training samples. A new design case can be predicted in a very short time by this well-trained model.

Module 4: Final Decision

Before beginning any design with the new emulator, it is necessary to justify the validity of the emulation model. A comparison between the results predicted by the emulator and its simulation with quantitative analysis is required.

1) Design-Adequate Decision

While training data sets are used for response analysis, the well-trained surrogate model can be applied to predict many new designs for deep case studies (e.g., further studies for detailed physical mechanisms, etc.) and design-adequate decision after model validation. Sometimes, multiple choices exist for design with the same or similar responses. With the surrogate model, this process can be quickly accomplished with predictions, and the results can be used to determine optimal design.

2) Quick Physics Survey for Complex Cases

Once the validation is confirmed, the emulator can be considered as a surrogate model to provide efficient prediction and Uncertainty Quantification (UQ) of turbulent flows in swirl injectors with any new geometries within the design space. This result can also help researchers complete a quick physics survey for complex cases, resulting in a short turnaround time.

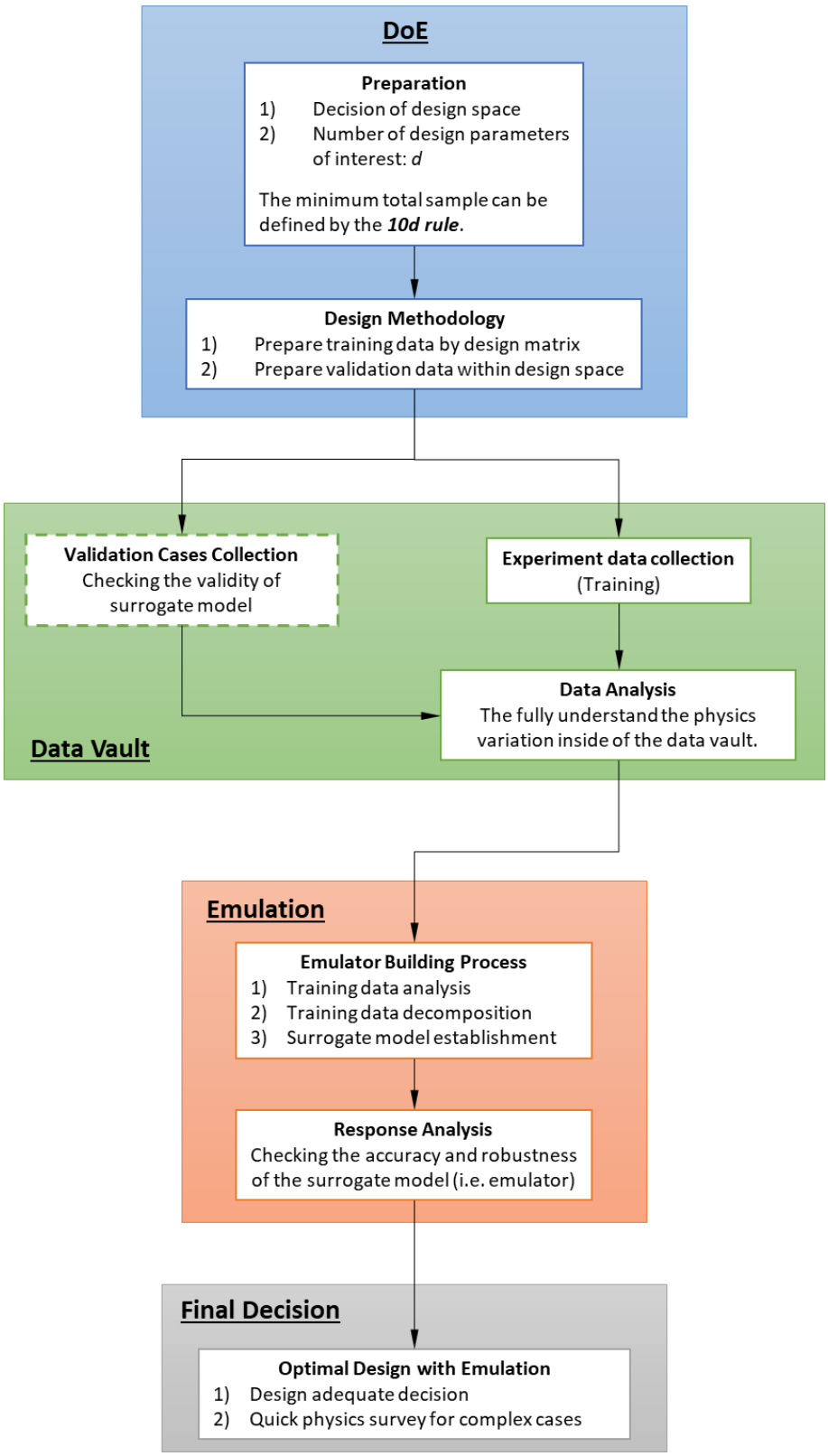

Figure 4—Schematic diagram for a new design strategy based on DoE and KSPOD

Above, the basic four modules for a new design are outlined. The common kernel-smoothed proper orthogonal decomposition (CKSPOD) based emulator used in this work requires only about 0.03 CPU hours on an Intel Xeon Processor E5-1650 V4 to predict the flowfield at a new design point for the problem which exploring spatiotemporal evolving flow emulation models presented in this study. The overall computation time, including data loading and training, is about 25 CPU hours (on an Intel Xeon Processor E5-1650 V4) to predict the flow evolution at a new design point for a time duration of 10 ms with 1,000 snapshots. For comparison, the corresponding LES calculation takes about 100,000 CPU hours (on the hexa-core AMD Opteron Processor 8431) for a spatial domain of over 40,000 numerical grid points. This strategy can be applied for any engineering design studies. If there are specific criteria for a target design, an optimization process can be further used with this surrogate model and specified conditions to determine the best design.

2.2 High-Fidelity Simulation Theory

2.2.1 Nonreacting Flow

The theoretical formulation for high-fidelity simulations is described in Zong and Yan [16] and Huo and Yang [6, 7], which treats supercritical fluid flows and combustion over the entire range of fluid thermodynamic states of concern. Turbulence closure is achieved by means of LES techniques. The effects of subgrid-scale motion are represented by the Smagorinsky eddy viscosity model. Thermodynamic properties are evaluated according to fundamental thermodynamic theories and a modified Soave-Redlich-Kwong equation of state. The Takahashi method, calibrated for high-pressure conditions, is

employed to obtain the mass diffusivity. Transport properties are evaluated using extended corresponding-state principles.

The numerical framework utilized to complete this research is based on a preconditioning scheme with a unified treatment of general-fluid thermodynamics [43, 44]. It applies a density-based, finite-volume methodology, along with a dual-time-step integration technique [45]. A second-order backward difference is used to accomplish temporal discretization, and a four-step Runge-Kutta scheme is applied to integrate the inner-loop pseudo-time term. A fourth-order central difference scheme in generalized coordinates is used to obtain spatial discretization. Fourth-order matrix dissipation is taken to assure numerical stability and minimum contamination of the solution. Lastly, a multi-block domain decomposition technique, associated with the message passing interface technique of parallel computing, is applied to optimize computation speed.

2.2.2 Supercritical Combustion Flow

The theoretical basis of the present study is described by Oefelein and Yang [46], who deal with supercritical fluid flows and combustion over the entire range of fluid thermodynamic states of concern. Turbulence closure is achieved using the LES technique. The Smagorinsky eddy viscosity model proposed by Erlebacher et al. [47] is employed to represent the effects of subgrid-scale motion. Thermodynamic properties, including density, enthalpy, and specific heat at constant pressure, are evaluated according to fundamental thermodynamic theories and the modified Soave-Redlich-Kwong equation of state. Transport properties, including thermal conductivity and dynamic viscosity, are estimated using an extended corresponding-state principle. Mass diffusivity is obtained by

the Takahashi method calibrated for high-pressure conditions [48]. The evaluation of thermodynamic and transport properties has been validated and implemented in previous studies [5, 49-51].

Modeling of interactions between turbulence and chemistry remains a critical issue. A precise classification of turbulent diffusion flame regimes has not been definitively created because diffusion flames do not have well-defined length, time, and velocity scales [52]. Local flame scales depend on local flow conditions. The chemical kinetics of kerosene combustion involve hundreds of species and thousands of elementary reaction steps, rendering the direct simulation of detailed chemistry computationally prohibitive. A steady laminar flamelet model is applied in the present study. The underlying assumption is that the local Damkohler number ($Da^{fl} = \tau_f/\tau_c$) is sufficiently large and the chemistry is sufficiently fast to follow the flow changes; here, τ_f and τ_c denote the flow characteristic time and chemical time, respectively. Unsteady effects and flame extinction occur when Da^{fl} is low. A prior study demonstrated that the local strain rate in the current flowfield is much smaller than the extinction strain rate (~107 s-1 at 250 atm) for oxygen/kerosene counterflow diffusion flames [51]. This implies that Da^{fl} is large and justifies the validity of the flamelet concept.

A three-component surrogate of kerosene [53], n-decane/n-propylbenzene/n-propylcyclohexane (74%/15%/11% by volume), has shown good agreement with the jet-stirred reactor data and is employed in this work. A skeletal mechanism with 106 species and 382 reactions, developed by Wang et al. [54], is implemented because of its computational efficiency and high accuracy in predicting global combustion characteristics

with a pressure range of 1-20 atm and an equivalence ratio range of 0.5-1.5. It is noted that the operating pressure for current simulations is 25.3 MPa. Further validation may be required to determine whether this skeletal mechanism is suitable at this extended pressure value. The solutions of counterflow diffusion flames are used to build the flamelet library, and it contains a set of equilibrium solutions in a range of strain rates at a pressure identical to that of LES simulations.

The numerical framework utilized in this work was established by implementing a preconditioning scheme and a unified treatment of general-fluid thermodynamics [43]. It employs a density-based, finite-volume methodology, along with a dual-time-step integration technique [45]. Temporal discretization is achieved using a second-order backward difference, and the inner-loop pseudo-time term is integrated with a four-step Runge-Kutta scheme. Spatial discretization is obtained using a fourth-order central difference scheme in generalized coordinates. Fourth-order matrix dissipation, developed by Swanson and Turkel [55], is taken to ensure numerical stability and minimum contamination of the solution. Finally, a multi-block domain decomposition technique, associated with the message passing interface technique of parallel computing, is applied to optimize computation speed.

CHAPTER 3. RESEARCH DESIGN

This chapter states the design process of this research. As the research framework mentioned in Chapter 2, there are four major steps of a design strategy while developing a new product, including DoE, creating a data vault, emulation, and final decision. This chapter dissects details of these four steps. In Section 3.1, it illustrates the geometric details of the swirl injector configuration and explains decisions regarding design parameters and design space. Both nonreacting flow and reacting-flow cases are included. Section 3.2 states the DoE methods chosen for this research. Section 3.3 elaborates and manifests three emulation models for the prediction of spatiotemporal evolving flow dynamic: common proper orthogonal decomposition (CPOD), kernel-smoothed proper orthogonal decomposition (KSPOD), and common kernel-smoothed proper orthogonal decomposition (CKSPOD). Section 3.4 addresses the data-driven framework for emulation result analysis.

3.1 Data Sets with Multiple Geometric Design Parameters

This dissertation conveys three major methodologies of emulation for spatiotemporal evolving flow. The order of the evolution process of these three methods is CPOD [1, 2], KSPOD [3], and CKSPOD. Each method requires a set of data to train the low-fidelity prediction model (kriging-based emulator). This section introduces the process of design parameter selection and the design matrix made by DoE.

3.1.1 Swirl Injector Configuration for Nonreacting Flow Cases

Figure 5 shows a schematic of the swirl injector being discussed in this dissertation. Liquid oxygen (LOX) is tangentially introduced into the injector and develops a swirling

film that is attached to the wall due to centrifugal force [10, 56]. Conservation of angular momentum results in a hollow gaseous core in the center region. The liquid film exits the injector as a thin conical sheet and subsequently undergoes atomization into droplets. The flow dynamics in this type of device under supercritical conditions have been extensively studied using LES techniques [11, 12].

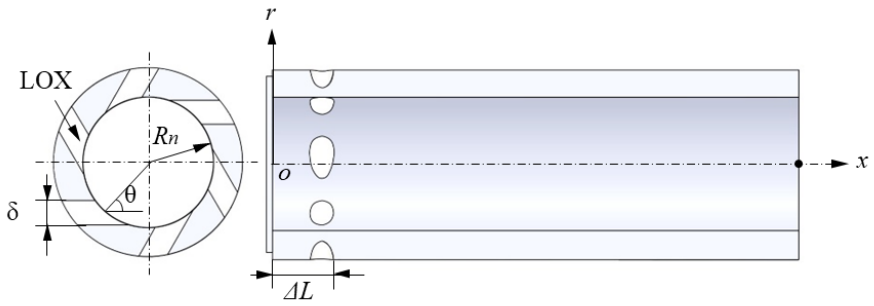

Figure 5—Schematic of swirl injector with five design parameters

In this research, the selection of design variables is dependent upon system requirements. The first emulation method developed for flow dynamics prediction is CPOD. In the first stage of research, five parameters that define the geometry of an injector are selected. The five parameters are injector length, L, injector radius, R_n, inlet slot width, δ, tangential inlet angle, θ, and the distance between the inlet and headend, ΔL. These design parameters are important in determining the injector performance, including the thickness, h, and spreading angle, α, of the liquid film at the injector exit. The selection of these design parameters is dependent upon engine requirements. Table 1 shows the design space and the ranges of each parameter considered in CPOD work.

To generalize the emulator framework, a broad range of these parameters was chosen. The range of injector lengths was chosen to include those of small upper-stage

rockets such as the RD-0110 [28], about 22.7 mm, and large first-stage engines like the RD-170 [29], about 93 mm.

Liquid oxygen (LOX) is delivered tangentially into the injector through inlets at a temperature of 120 K. The operating pressure is 100 atm, typical of contemporary liquid rocket engines. The ambient gas is oxygen at 300 K. The flow dynamics of this class of injectors have been systematically investigated in detail by Zong et al. [11] and Wang et al. [12]. First, a set of high-fidelity simulations is conducted based on conditions in the design space described in Table 1, then the common flow structures for surrogate modeling are extracted.

Table 1—Design space for injector geometric parameters (30 sets for CPOD [2])

L (mm)	R_n (mm)	θ (°)	δ (mm)	ΔL (mm)
20-100	2.0-5.0	45-75	0.5-2.0	1.0-4.0

In CPOD research [1, 2], sensitivity analysis is used to identify the most significant parameters dictating injector performance. The sensitivity analysis shows that the injection width, δ, is the most important parameter in the determination of the spreading angle of the liquid film. The tangential inlet angle, θ, and the injection width, δ, significantly affect the liquid-film thickness, while the injector length, L, and radius, R_n, play minor roles. Thus, both KSPOD and CKSPOD work focuses on the injection width, δ, angle, θ, and distance between the inlet and headend, ΔL.

Table 2 tabulates baseline geometry and operating conditions, including the LOX inlet temperature, T_{in}, ambient temperature, T_∞, ambient pressure, p_∞, and mass flow rate,

ṁ. Another parameter is the geometric constant, K, a nondimensional parameter that can be used to evaluate the flow characteristics of swirl injectors, namely the liquid-film thickness and spreading angle. The former controls film atomization and the latter mixing efficiency [9]. The geometric constant takes the following definition,

$$\mathrm{K} = A_n R_{in} / A_{in} R_n, \tag{1}$$

where A_n denotes the cross-sectional area of the injection exit and A_{in} the total inlet area. The geometric constant is an indicator of the swirl strength. When the value is high, a large angular momentum is present in the liquid film, leading to a wide spreading angle.

Table 2—Baseline geometry and operating conditions.

R (mm)	R_{in} (mm)	L (mm)	$\dot{m}$ (kg/s)	T_{in} (K)	T_∞ (K)	p_∞ (MPa)
4.50	0.85	25	0.17	120	300	10

Table 3 shows the design space and the range for each design variable. The distance between the inlet and headend, ΔL, is decided by a rule of thumb to be 1.5-2 times the injector width [9]. This is an optimal location, determined from a trade-off study, for avoiding (1) excessive viscous losses when the injection slit is too close to the headend and (2) low-frequency oscillations due to the presence of a large recirculation zone if the inlet is too far from the headend. The design spaces of injection width, δ, and angle, θ, are decided by the desired range of spreading angle (50-62˚) and film thickness (0.66-1.50 mm). With these numbers and the geometric constant, the ranges of δ and θ can be estimated.

Table 3—Design space for second (30 sets for KSPOD & CKSPOD [3]).

Design Variable	θ (deg)	δ (mm)	ΔL (mm)
Design Range	35.0-62.2	0.27-1.53	0.85-3.40

3.1.2 Swirl Injector Configuration for Supercritical Reacting-Flow Cases

Figure 6 plots the gas-liquid jet-swirl injector being investigated, mimicking the RD-170 engine, which was designed and produced by NPO Energomash in the Soviet Union and used to power the Energia launch vehicle. The injector consists of four parts: inner jet, outer swirler, recess region, and taper region. Gaseous Oxygen (GOX) is injected axially in the inner jet, while liquid kerosene is tangentially introduced into the coaxial swirler. The process of mixing GOX and kerosene initiates in the recess region and intensifies in the taper region and downstream of the injector. The geometric parameters of these dimensions are listed in Table 4. The recess length has been identified to have significant effects on the mixing characteristics for rocket injectors in many previous studies. In this disseration, six cases with different recess lengths (L_r) are considered in a range of 0-16 mm to explore the change flow and flame dynamics. The total axial length of the annulus upper surface is fixed at 16 mm; the length of the annulus lower surface changes (shielding, L_s) accordingly when the recess length varies. Table 5 displays recess length and shielding length for the six cases. Recess length decreases with increasing case number. Case 3, with a recess length of 5.5 mm, is the baseline, while Case 1 is fully recessed and Case 6 has no recess region.

To isolate the effect of recess length, the operating conditions for all cases are identical and listed in Table 6; $\dot{m}_o$ and $\dot{m}_f$ represent the injection mass flow rate of oxidizer and fuel, respectively. $T_{in,o}$, $T_{in,f}$, and p_a denote the oxidizer injection temperature, fuel injection temperature, and ambient pressure, respectively. Given the information on the operating conditions and geometric parameters, the nominal momentum flux ratio between oxidizer and fuel streams is estimated as $\rho_o U_o^2/\rho_f U_f^2$. The reference density of oxidizer and fuel is 131 and 640 kg/m3, respectively. The reference velocity of oxidizer stream is calculated as 102 m/s, while the velocity of kerosene stream needs to be evaluated carefully based on the shield length. For Case 1, without shielding, kerosene radially penetrates the axial GOX stream, and thus U_f refers to the radial velocity component at the inlet, 24.5 m/s. For other cases with shields, U_f is most appropriately represented by the axial velocity component in the outer swirler, 26.6 m/s. The momentum flux ratio is thus obtained as 3.5 for Case 1 and 3.0 for Cases 2-6.

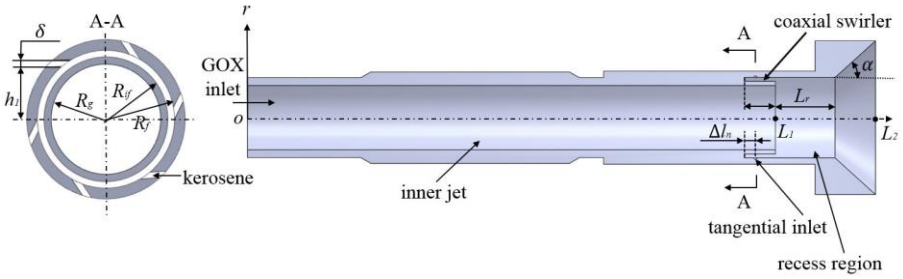

Figure 6—Geometry of RD170 GOX/kerosene jet-swirl injector (baseline design)

Table 4—Geometric parameters of the jet-swirl injector

δ (mm)	h_1 (mm)	R_g (mm)	R_{if} (mm)	R_f (mm)	α (°)
0.665	5.75	5.62	6.365	7.03	42
L_1 (mm)	L_2 (mm)	L_f (mm)	L_r (mm)	Δl_n (mm)	
93	113.1	5.5	10.5 (baseline)	2	

Table 5—Lengths of recess and shielding for training and validation (test) cases

Cases	1 (none)	2	3 (baseline)	4	5	6 (full)	7 (test)	8	9	10
L_s (mm)	0	3.0	5.5	9.0	12.5	16.0	7.25	4.25	10.75	8.125
L_r (mm)	16.0	13.0	10.5	7.0	3.5	0	8.75	11.75	5.25	7.875

Table 6—Operating conditions of all jet-swirl injector cases

	Oxidizer	Fuel
Mass flow rate, kg/s	1.33	0.477
Static pressure, bar	252.96	252.96
Temperature, K	687.7	492.2
Inlet velocity, m/s	186.69	U_r = 24.53 U_θ = 49.06

The computational domain consists of the injector interior ($18R_g$ in axial direction) and a downstream region ($25R_g$ and $7R_g$ in the axial and radial directions, respectively). A high-fidelity simulation of the entire domain described above is computationally prohibitive; a cylindrical sector is thus considered in the present work with the periodic boundary condition specified in the azimuthal direction. Acoustically non-reflecting

boundary condition [57] is implemented at the inlet of the inner jet. The downstream boundary in both axial and azimuthal directions is treated by a sponge-layer method. No-slip and adiabatic boundary conditions are applied at the injector solid surfaces. A reference pressure is applied to preserve the average pressure in the computational domain.

3.2 Design of Experiment (DoE)

3.2.1 Maximum Projection (MaxPro)

The DoE methodology is a statistical approach for careful selection of input variables for a given design space; it facilitates the design process and reduces the number of total sample points required to efficiently explore the design space. Given the design space in Table 1, if ten variations are assigned for each design parameter, the total number of design points is 105 for a traditional full-factorial design. It is impractical to perform so many simulations, due to the extensive computing resources required to acquire usable data. A DoE methodology is therefore required to reduce the number of design points and still capture the prominent features in the design space. To this end, the maximum projection (MaxPro) design proposed by Joseph et al. [39] is implemented for excellent space-filling properties and GP modeling predictions. Thirty points in the expected range of 5-10d (e.g., 6d rule with $d = 5$, the number of design parameters) points as suggested by Loeppky et al. [13], which is commonly used in computer experiment literature, are simulated over the entire design space. The accuracy of prediction should always be checked to determine whether additional simulations are needed (see Loeppky et. al. [13]). Figure 7 shows a two-dimensional projection of the 30 simulation runs by MaxPro design,

which distributes representative design points to fill the two-dimensional projection of the design space. Good space-filling properties are observed for all parameters.

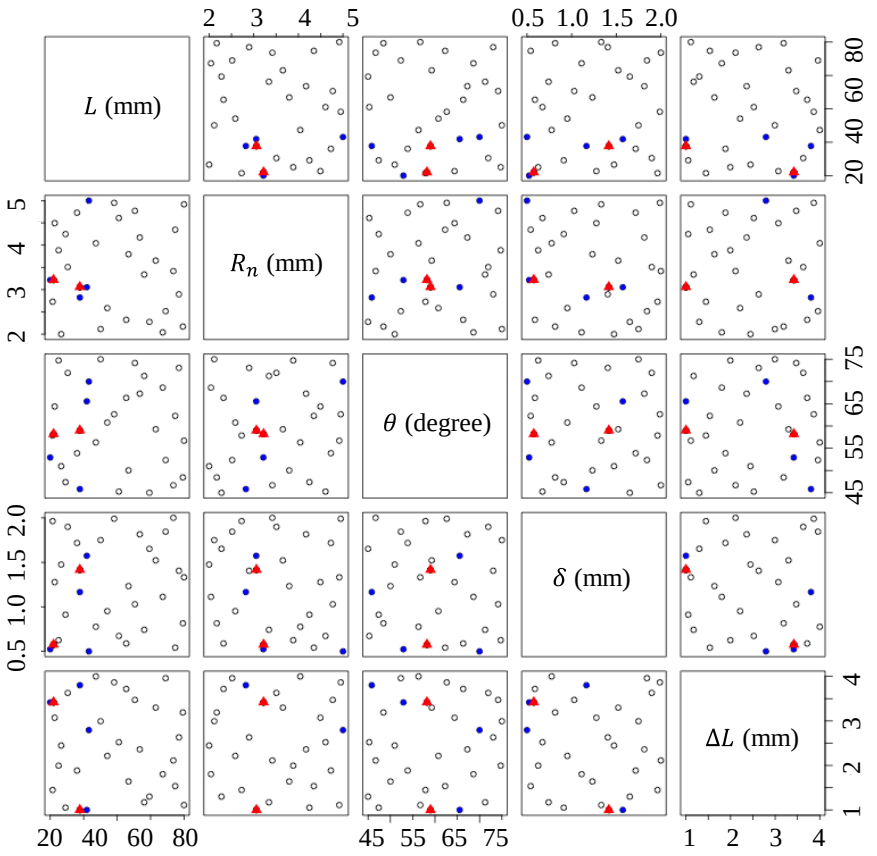

Figure 7—Two-dimensional projections of design points: benchmark points (▲) and baseline and neighboring points (●)

3.2.2 Sliced Latin Hyper Cube Design (SLHD)

With prior knowledge of the major contributing geometric parameters, sample size in KSPOD and CKSPOD analysis is determined based on the $10d$ rule of thumb [13], where d denotes the total number of design parameters. Because of the sensitivity analysis resulting from CPOD analysis, the second stage of emulation study considers three design variables (θ, δ, and ΔL). Although MaxPro provides good space-filling properties on both

the design space and its projections [39], it does not provide a sequential design capability, giving optimal space-filling performance in batches of DoE. To that end, Sliced Latin Hypercube Design (SLHD) [58, 59] is selected. In SLHD, the space-filling performance of the design points in each slice is optimal. The overall design matrix contains five slices, and each slice includes six design points.

Figure 8 shows the two-dimensional projections of the design points categorized by different slices. Each case requires about 100,000 CPU hours for high-fidelity simulation to obtain statistically significant data. A total of 1,000 snapshots spanning 10 ms are acquired after the flowfield reaches its stationary state (~12 ms). The snapshots are subsampled every 20 temporal iterations, each with a time step of $0.5\mu s$. A temporal resolution of 50 kHz is achieved, according to the Nyquist criterion.

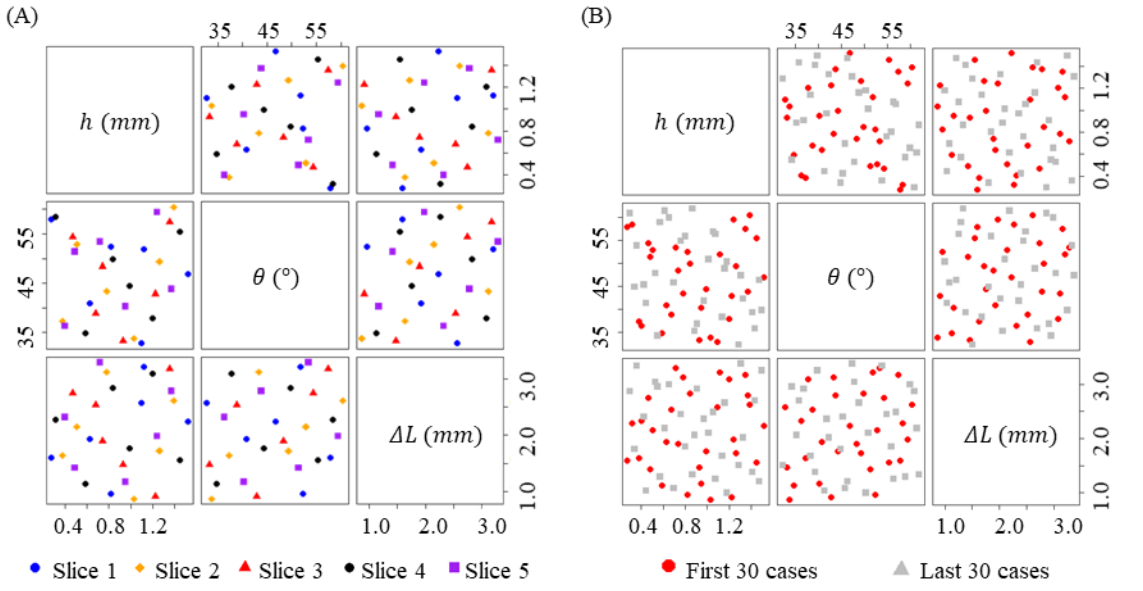

Figure 8—2D projection of design points obtained by SLHD methodology: (A) the first 30 cases for emulation training data in five slices; (B) the first 30 cases (red) and the last 30 cases (gray) in the design space

3.3 Kriging Surrogate Models (Emulators)

This section introduces three emulator models, CPOD, KSPOD, and CKSPOD, and subsequently presents the proposed emulator model and a parallelized algorithm for parameter estimation. A key theme in this section (and indeed, this dissertation) is the elicitation and incorporation of flow physics within the emulator model. This not only allows for efficient and accurate flow predictions through simplified model assumptions (a low-fidelity model) but also provides a data-driven method for extracting useful flow physics, which can guide future experiments. As demonstrated in Section CHAPTER 3, both objectives can be achieved despite limited runs and complexities inherent in flow data. Table 7 summarizes the elicited flow physics and the corresponding emulator assumptions; each point is discussed in greater detail below.

Table 7—Elicited flow physics and corresponding assumptions for the emulator model

FLOW PHYSICS	MODEL ASSUMPTIONS
COHERENT STRUCTURES IN TURBULENT FLOW (LUMLEY, 1967 [60])	POD-based kriging
SIMILAR REYNOLDS NUMBERS FOR COLD-FLOW (STOKES, 1851 [61])	Linear-scaling modes in CPOD
DENSE SIMULATION TIMESTEPS	Time-independent emulator
COUPLINGS BETWEEN FLOW VARIABLES (POPE, 2001 [62])	Co-kriging framework with covariance matrix $\mathbf{T}$
FEW-BUT-SIGNIFICANT COUPLINGS (POPE, 2001 [62])	Sparsity on $\mathbf{T}^{-1}$

3.3.1 Common Proper Orthogonal Decomposition (CPOD)

The full emulator model and algorithm are provided in the companion statistical paper [1], which considers the statistical properties of a broader class of models. This dissertation focuses on applying new machine-learning techniques and investigates the practical performance of the emulator with respect to flow physics. Part of the emulator characteristics and related UQ algorithms will be addressed in Section 3.4.3.

A brief overview of POD is first provided by Lumley in 1967 [60]. For a fixed injector geometry, $f(\boldsymbol{x}, t)$ denotes a flow variable (e.g., pressure) at spatial coordinate $\boldsymbol{x} \in \mathbb{R}^2$ and flow time t. The following decomposition of $f(\boldsymbol{x}, t)$ into separable spatial and temporal components is provided by POD:

$$f(\boldsymbol{x}, t) = \sum_{k=1}^{\infty} \beta_k(t)\phi_k(\boldsymbol{x}), \tag{2}$$

with the spatial eigenfunctions, $\{\phi_k(\boldsymbol{x})\}_{k=1}^{\infty}$, and temporal coefficients, $\{\beta_k(t)\}_{k=1}^{\infty}$, given by

$$\phi_k(\boldsymbol{x}) = \underset{\substack{\|\psi\|_2=1, \\ \langle\psi,\phi_l\rangle=0,\forall l<k}}{\operatorname{argmax}} \int \{f(\boldsymbol{x}, t)\psi(\boldsymbol{x})d\boldsymbol{x}\}^2 dt \tag{3}$$

$$\beta_k(t) = \int f(\boldsymbol{x}, t)\phi_k(\boldsymbol{x})d\boldsymbol{x}. \tag{4}$$

Following Berkooz et al. (1993) [15], $\{\phi_k(\boldsymbol{x})\}_{k=1}^{\infty}$ is referred to as the spatial POD modes for $Y(\boldsymbol{x}, t)$, and its corresponding coefficients $\{\beta_k(t)\}_{k=1}^{\infty}$ are referred to as time-varying coefficients. There are two key reasons for choosing POD over other reduced-basis models. First, one can show [63] that any truncated representation in Equation (2) gives the best flow reconstruction of $f(\boldsymbol{x}, t)$ in L_2-norm, compared to any other linear expansion

of space/time products with the same number of terms. This property is crucial for our application, since it allows the massive simulation data to be optimally reduced to a smaller training dataset for the proposed emulator. Second, the POD has a special interpretation in terms of turbulent flow. In the seminal paper by Lumley (1967), it is shown that, under certain conditions, the expansion in Equation (2) can extract physically meaningful coherent structures that govern turbulence instabilities. For this reason, physicists use POD as an experimental tool to pinpoint key flow instabilities, simply through an inspection of $\phi_k(\boldsymbol{x})$ and the dominant frequencies in $\beta_k(t)$. For example, using POD analysis, Zong and Yang (2008) [11] have shown that the two flow phenomena, hydrodynamic wave propagation on LOX film and vortex core excitation near the injector exit, are the key mechanisms driving flow instability. This is akin to the use of principal components in regression, which can yield meaningful results in applications where such components have innate interpretability.

Unfortunately, POD is only suitable for extracting instability structures at a single geometry, whereas for emulation, a method is needed that can extract common structures over varying geometries. Therefore, a new decomposition, CPOD, is proposed here. The key assumption of CPOD is that, under a physics-guided partition of the computational domain, the spatial distribution of coherent structures scales linearly over varying injector geometries. For cold flows, this can be justified by similar Reynolds numbers (a nondimensional number characterizing flow dynamics) over different geometries [61]. This is an example of model simplification through elicitation; such a property likely does not hold for general flows. This linearity assumption is highly valuable for computational efficiency, because flows from different geometries can subsequently be rescaled onto a

common spatial grid for instability extraction. Figure 9 visualizes this procedure. The grids for each simulation are first split into four parts: from injector headend to inlet, from inlet to nozzle exit, and the top and bottom portions of the downstream region. Each part is then proportionally rescaled to a common reference grid according to changes in the geometric variables L, R_n, and ΔL (see Figure 5). From a physics perspective, such a partition is necessary for the linearity assumption to hold.

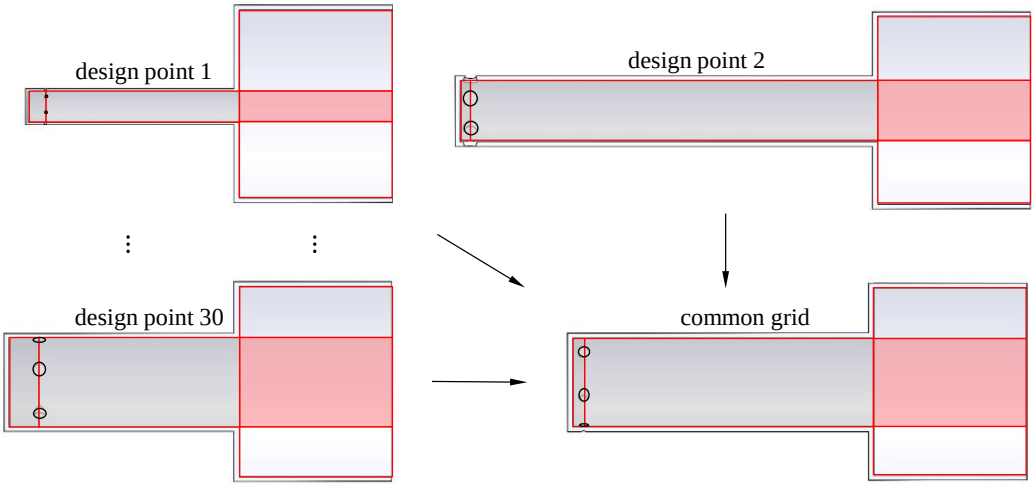

Figure 9—Schematic of the common-grid generation process

The primary objective of CPOD study is to develop an emulator model that uses data from 30 simulation runs to predict the flowfield of a new design point within a practical turnaround time. Utilizing the tools described above—the sensitivity analysis for parameter screening and the decision tree for partitioning the design space into jet-swirl cases—a surrogate model for flowfield emulation is proposed. The kriging surrogate model combines machine-learning techniques, statistical modeling, and a physics-driven data-reduction method. A brief explanation for each part of this model is provided below,

followed by a discussion of the specific mathematical details. A complete description of the model development from the statistical perspective is given in Mak et. al. [1].

First, the proposed model is constructed through a POD analysis of the simulation dataset used for training. For a given flow property, f, the POD analysis determines a set of orthogonal basis functions, ϕ_j , such that the projection of the property onto these basis functions has the smallest error, defined as $E\left(\left\|f-\hat{f}\right\|^2\right)$, where $E(\cdot)$ and $\|\cdot\|$ denote the time average and norm in the L^2 space, respectively [60]:

$$\hat{f}(\boldsymbol{x},t)=\sum_{j=0}^{n}\beta_j(t)\phi_j(\boldsymbol{x}). \tag{5}$$

The basis functions, or mode shapes, are spatial distributions of the fluctuating fields of flow properties, which can be closely linked to physical phenomena and coherent structures. The basis functions are ordered so that the lowest modes have the highest "energy," as defined by the inner product of f. The flow properties for POD analysis include pressure, density, temperature, and velocity components. POD decomposition yields not only the eigenfunction modes, ϕ_j, but also their corresponding time-varying coefficients, β_j, which are referred to as POD coefficients. It should be noted that this process is not completed for the entire dataset; physical variables are processed separately. To treat the data together, scaling and dimensions must be carefully formulated to obtain interpretable mode shapes.

While the usage of POD simplifies the complex nature of a spatiotemporal model, a common set of basis functions is required for the emulator to accommodate different

injector geometries. Physically, this entails the extraction of a common set of coherent structures over the design space; one option is to select a computational region of interest that is unaffected by any design changes [64]. Building upon the basis functions generated by the POD analysis, an emulator can be obtained when a set of common basis functions exist. The wide disparity of geometries in the design space, as illustrated in Figure 5, represents a challenge for the current study.

As mentioned above, this work utilizes a common grid for the 30 grid systems to find a set of common basis functions. To achieve this, the densest grid system among all cases, which has the highest number of cells, is identified and split into four sections covering the effects of design parameters on the simulated grid. This partitioned grid is used for interpolation and rescaling each simulated case to obtain a common grid. An inverse distance weighting interpolation method with ten nearest neighborhood points is then employed to map the original raw data onto the common grid [65]. Algorithmically, the CPOD expansion is obtained by first rescaling the different cases to the common grid, computing the POD expansion, and finally rescaling the resulting modes back to the original grid [1].

Because of the limited variation of the Reynolds number among the different injector geometries, the scaling of the data to the common grid is appropriate in the present study. The smallest injector diameter of concern is 4 mm, with a corresponding exit velocity of 27.5 m/s. With a LOX density of 1,000 kg/m^3 and viscosity of 0.114 cP, the Reynolds number based on the injector diameter is approximately 9.6×10^5. The largest injector diameter in the design space has a value of 10 mm with a corresponding exit velocity of 11 m/s. At the same operating condition, the Reynolds number is approximately

9×10^5 for all designed experiments. For some geometries, where the liquid film does not produce a noticeable spreading angle, the Reynolds number is reduced to approximately 9×10^5. Despite this difference, the model can avoid excessive smoothing, provided that the correlation function is bounded correctly. This scaling of POD modes to establish common basis functions is vital to the emulator. It should be noted that the scaling is only appropriate for flow simulations that do not exhibit distinctively different physical phenomena, such as those of reacting-flow simulations, where the mode shapes change drastically. Additional similarity parameters may be necessary when different physics and chemical reactions are incorporated, as reviewed by Dexter et al. [66].

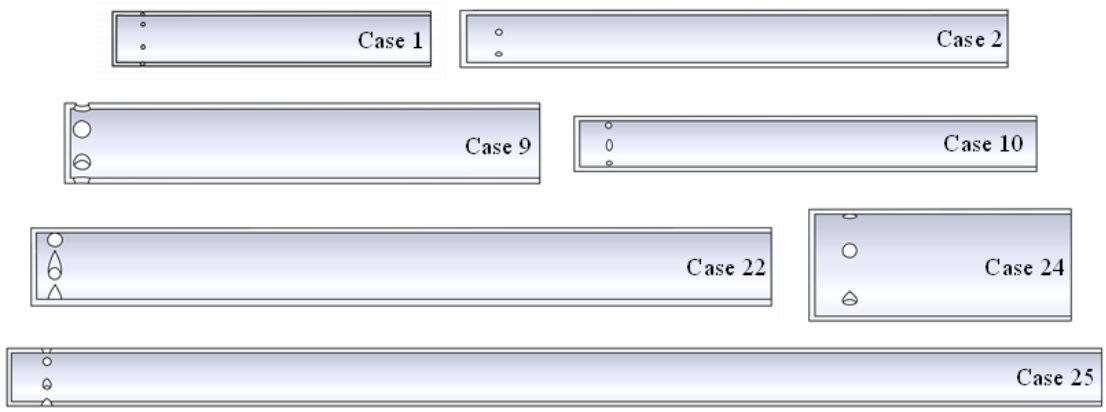

Figure 10—Schematics of the different injector geometries in the design space

The mathematical details for CPOD are provided below [1]. It is supposed that n simulations are conducted at various design geometries $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n$ and $f(\boldsymbol{x}, t; \boldsymbol{c}_i)$ is the simulated flowfield at design $\boldsymbol{c}_i$ for a given time t and spatial coordinate $\boldsymbol{x}$. The k-th CPOD mode is defined as

$$\phi_k(\boldsymbol{x}) = \underset{\psi:\|\psi\|_2=1}{\operatorname{argmax}} \sum_{i=1}^{n} \int \left[\int \mathcal{M}_i[\psi(\boldsymbol{x})] f(\boldsymbol{x}, t; \boldsymbol{c}_i) d\boldsymbol{x} \right]^2 dt\,,$$
$$s.t. \int \psi(\boldsymbol{x}) \phi_l(\boldsymbol{x}) d\boldsymbol{x} = 0, \forall\, l < k. \tag{6}$$

Here, the map $\mathcal{M}_i$: $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ is the transformation that linearly scales spatial features from the common geometry $\boldsymbol{c}$ to the *i*-th geometry $\boldsymbol{c}_i$. The sequence of POD coefficients is defined as

$$\beta_k(\boldsymbol{c}_i, t) = \int \mathcal{M}_i\{\phi_k(\boldsymbol{x})\} f(\boldsymbol{x}, t; \boldsymbol{c}_i) d\boldsymbol{x}, \tag{7}$$

with the corresponding POD expansion using *K* modes given by

$$f^{(K)}(\boldsymbol{x}, t; \boldsymbol{c}_i) = \sum_{k=1}^{K} \beta_k(\boldsymbol{c}_i, t) \mathcal{M}_i\{\phi_k(\boldsymbol{x})\}. \tag{8}$$

The transformation allows for the extraction of common basis functions. In addition, the obtained modes can be used to identify key mechanisms of flow dynamics. It should be noted that reacting-flow simulations are characterized by additional dimensionless parameters, and linear mapping may not perform well when combustion is involved.

Two computational challenges must be addressed to implement this methodology. As previously mentioned, to calculate the inner product of the snapshots from different simulation cases, a common set of spatial grid points is needed. Not only does the calculation of the inner product become a computational bottle-neck, as the covariance matrix consists of snapshots from each simulation, the number of modes required to capture

a certain energy level is significantly increased relative to an individual simulation, which can capture more than 95% of the total energy with approximately six modes [11]. The computation of CPOD modes and associated time-varying coefficients requires eigen-decomposition of a $nT \times nT$ matrix, where n is the number of simulation cases and T is the number of snapshots. This usually requires $O(n^3T^3)$ computation work. A typical value for T is 1,000 snapshots spanning 10 ms, which achieves a frequency resolution of 100 Hz. An iterative method of eigen-decomposition based on periodic restarts of Arnoldi decompositions is then used here to quickly calculate the first few eigen-vectors with the largest eigenvalues. These eigenvalues can also be interpreted as the amount of "energy," as defined by the inner product used to calculate the covariance matrix. For a particular data reconstruction using a linear combination of POD modes and associated time-varying coefficients, there is reconstruction error, which decreases when more eigen-vectors, or POD modes, are included.

After the data decomposition step, a kriging model is applied to the CPOD time-varying coefficients $\beta_k(\boldsymbol{c}_i, t)$. With the mean and variance computable in closed form, UQ and confidence intervals (CI) can be calculated easily. The mathematical approach of kriging is described here. For notational simplicity, let $\beta(\boldsymbol{c})$ denote $\beta_k(\boldsymbol{c}, t)$, the k-th CPOD coefficient at setting $\boldsymbol{c}$ and time step t. As the temporal resolution is fine, there is no practical need to estimate temporal correlations, especially because predictions will not be made between timesteps. This time-independent emulator uses independent kriging models at each instant of time, assuming the following GP model:

$$\beta(\boldsymbol{c}) = \mu + Z(\boldsymbol{c}), \qquad Z(\boldsymbol{c}) \sim N\{0, \sigma^2 R(\cdot,\cdot)\}. \tag{9}$$

Here, μ is the mean, $Z(\boldsymbol{c})$ is a zero-mean GP with variance σ^2, and $R(\cdot,\cdot)$ is a pre-specified correlation function governed by unknown parameters $\eta's$. A typical choice for $R(\cdot,\cdot)$ is the Gaussian correlation function

$$R(\boldsymbol{c}_i, \boldsymbol{c}_j) = \exp\left[-\sum_{k=1}^{p} \eta_k \left(c_{ik} - c_{jk}\right)^2\right] \tag{10}$$

where p is the number of input parameters.

Next, it is supposed that the function values $\boldsymbol{\beta}^{(n)} = [\beta(\boldsymbol{c}_i)]_{i=1}^{n}$ are observed at input settings $\{\boldsymbol{c}_i\}_{i=1}^{n}$ and that $\boldsymbol{c}_{new}$ is a new setting for which prediction is desired. Conditional on the observed values $\boldsymbol{\beta}^{(n)}$, the best linear unbiased estimator of $\beta(\boldsymbol{c}_{new})$ can be shown to be [67]

$$\hat{\beta}(\boldsymbol{c}_{new}) = \mu + \boldsymbol{r}_{new}^T \boldsymbol{R}^{-1}\left(\boldsymbol{\beta}^{(n)} - \mu\mathbf{1}\right). \tag{11}$$

Here, $\mathbf{1}$ is the n x 1 vector of ones, $r_{new} = [R(\boldsymbol{c}_i, \boldsymbol{c}_{new})]_{i=1}^{n}$ is the n x 1 vector of correlations between the new point and sampled points, and $\boldsymbol{R} = \left[R(\boldsymbol{c}_i, \boldsymbol{c}_j)\right]_{i=1}^{n}{}_{j=1}^{n}$ is the covariance matrix for the sampled points. Such a predictor minimizes the mean-squared prediction error, a commonly-used criterion for prediction error. In the context of flowfield prediction, employing this kriging estimator allows for obtaining accurate flow predictions from the CPOD coefficients. It can also be shown [67] that this best mean-squared prediction error predictor is unbiased, matching the expected and true function values.

To close the formulations, the model parameters μ, σ^2, and η must be trained using data. A technique called maximum-likelihood estimation (MLE), a ubiquitous estimation

technique in statistical literature [68], is employed. The key concept in MLE is to discover the optimal parameter setting that minimizes the likelihood function of the GP model. In the present work, optimization is achieved by means of the L-BFGS algorithm [69], a method employed for many training algorithms. A more detailed explanation can be found in Santner et al. [67].

In CPOD emulation process, the kriging models are trained independently over each time step, due to the inherent fine-scale temporal resolution of the simulation. This time-independence assumption is made for two reasons. First, the fully developed flow is treated as statistically stationary and has high-frequency resolution, so no practical value exists for estimating temporal correlations. Second, as in the high-fidelity simulation procedure, the assumption of time-independence allows the exploitation of parallel computation in training the emulator model. Once the model is trained, the predictor is used with the CPOD expansion to predict the flow evolution at a new design point:

$$\hat{f}(\boldsymbol{x}, t; \boldsymbol{c}_{new}) = \sum_{k=1}^{K} \hat{\beta}_k(\boldsymbol{c}_{new}, t)\mathcal{M} \quad \{\phi_k(\boldsymbol{x})\}. \tag{12}$$

It is worth noting that the computation time of the proposed model is orders of magnitude smaller than that of LES. Simulation data that typically takes a week, or around 30,000 CPU hours, to acquire can be predicted by the model with an associated uncertainty in under an hour. The procedure of CPOD is stated in Algorithm 1.

Algorithm 1—Common proper orthogonal decomposition (CPOD)

DATA:	For each design setting in $\{\boldsymbol{x}_i\}_{i=1}^{H}$, the flow evaluation at each spatial location and time step $f(\boldsymbol{x}_i, \boldsymbol{u}_j, t_q)$ is provided, where $\{\boldsymbol{u}_j\}_{j=1}^{K}$ is the spatial location and $\{t_q\}_{q=1}^{m}$ is the time step. $\boldsymbol{x}_i \in \mathbb{R}^{n\times m}$ and $n \gg m$.	
TRAINING:	*Step 1:*	Concatenate all H set of training cases along the time direction and create a huge matrix $\boldsymbol{X} \in \mathbb{R}^{n\times mH}$. Then, calculate the CPOD covariance by $\mathcal{C} = \frac{\boldsymbol{X}^T\boldsymbol{X}}{n-1}$ and $\mathcal{C} \in \mathbb{R}^{mH\times mH}$. Proper orthogonal decomposition (POD) based on singular-value decomposition (SVD) with $\mathcal{C}$ is performed and can be written as $f(\boldsymbol{X}, \boldsymbol{U_j}, T_Q) = \sum_{k=1}^{K} \hat{\beta}^k(\boldsymbol{X}, T_Q)\tilde{\phi}^k(\boldsymbol{X}, \boldsymbol{U_j})$ for the first dominated K mode, $T_Q = \{t_{1,q}^T, \dots, t_{H,q}^T\}^T$, and $t_q \in \mathbb{R}^{m\times K}$ Note: $\hat{\beta}^k(\boldsymbol{X}, T_Q) \in \mathbb{R}^{mH\times K}$ and $\tilde{\phi}^k(\boldsymbol{X}, \boldsymbol{U_j}) \in \mathbb{R}^{n\times K}$
	Step 2:	For each time step, t_q, and each mode, k, perform an ordinary kriging model on $\{\tilde{\beta}^k(\boldsymbol{X}, t_{1,q}), \dots, \tilde{\beta}^k(\boldsymbol{X}, t_{H,q})\}$ with inputs $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_H\}$ and the predictive function at an untried setting, x_{new}, $\hat{\beta}^k(\boldsymbol{x}_{new}, t_q)$.
	Step 3:	Since all training cases are concatenated together to create a huge matrix, $\boldsymbol{X} \in \mathbb{R}^{n\times mH}$, for CPOD covariance, all training cases share the same set of POD modes, $\tilde{\phi}^k(\boldsymbol{X}, \boldsymbol{U_j}) \in \mathbb{R}^{n\times K}$, which is also applied for the new prediction case.
PREDICTION:	At an untried setting x_{new}, compute $\hat{\beta}^k(\boldsymbol{x}_{new}, t_q), q = 1, \dots, m; k = 1, \dots, K$, and $\tilde{\phi}^k(\boldsymbol{X}, \boldsymbol{U_j}), j = 1, \dots, J; k = 1, \dots, K$, then $f(\boldsymbol{x}_{new}, \boldsymbol{U_j}, t_q) = \sum_{k=1}^{K} \hat{\beta}^k(\boldsymbol{x}_{new}, t_q)\tilde{\phi}^k(\boldsymbol{X}, \boldsymbol{U_j})$, where $j = 1, \dots, J$ and $q = 1, \dots, m$.	

3.3.2 Kernel-Smoothed Proper Orthogonal Decomposition (KSPOD)

This section introduces the idea of KSPOD, which combines statistical modeling with a data-reduction method to obtain a reduced-basis model. The proposed KSPOD method can be viewed as a generalization of POD used for flow emulation; POD decomposes the flowfield into an expansion consisting of spatial eigenfunctions, called POD modes, and corresponding time-varying coefficients. Such a decomposition can be written in the following form:

$$f(\boldsymbol{x}, t) = \sum_{k=1}^{\infty} \beta_k(t)\phi_k(\boldsymbol{x}). \tag{2}$$

The approximation form is

$$f(\boldsymbol{x}, t) \approx \sum_{k=1}^{K} \beta^k(t)\phi^k(\boldsymbol{x}), \tag{13}$$

where $f(\boldsymbol{x}, t)$ is the simulated flowfield at spatial location $\boldsymbol{x}$ and time t, and $\beta^k(t)$ and $\phi^k(\boldsymbol{x})$ represent the time-varying coefficient and basis function for the i-th mode, respectively. As indicated in Equation (13), the expansion is typically truncated at the first K terms, where K is chosen such that the reconstructed flowfield retains a desired degree of accuracy. In practice, the time-varying coefficients and basis functions are obtained through an eigen-decomposition of the inner product of a flowfield variable [15]. Equation (13) can be viewed as the optimal decomposition of $f(\boldsymbol{x}, t)$ using a basis expansion of K terms.

From a physics perspective, POD provides valuable insights regarding the important physics present in the flowfield. The basis function, or mode shape $\phi^k(\boldsymbol{x})$, can be interpreted as spatial distributions of the fluctuation field (e.g., pressure, density, temperature, and velocity components); it represents the dominant coherent structure, such as acoustic waves in the system [70]. A spectral analysis of the POD coefficients can be performed to identify flow periodicity and characteristic frequencies for hydrodynamic and acoustic instabilities. The index for the basis expansion in Equation (13) is determined by the rank of the energy content in the eigen-decomposition calculation and provides insight regarding which flow structure is more important. The first few terms in the expansion represent more energy-containing structures, and the remaining terms represent increasingly weaker flow features.

In CPOD, the emulator cannot predict detailed structures of spatiotemporal evolving flow. The goal of building an improved emulator is to employ the flow features extracted using POD within a statistical framework, allowing the training of an emulator for flow prediction. To this end, the popular machine-learning technique GPR, kriging, is employed to predict POD modes and time-varying coefficients at a new design setting. Below, a brief introduction into the mathematical formulation behind kriging is provided, followed by a description of how such a model is incorporated into the KSPOD framework.

Kriging is a powerful learning technique that leverages a GP-based model to learn the structure of an unknown function by sampling this function at specific points. A previously defined mathematical framework for kriging is followed [71, 72]. It is assumed that the unknown function of interest, $Y(\boldsymbol{c})$, $\boldsymbol{c} \in \mathbb{R}^d$, is a realization from the stochastic process

$$Y(\boldsymbol{c}) = \mu + Z(\boldsymbol{c}), \tag{14}$$

where $\boldsymbol{x}$ is a d-dimensional vector, μ is the mean of the process, and $Z(\boldsymbol{c})$ is a zero-mean GP with $Var\{Z(\boldsymbol{c})\} = \sigma^2$ and correlation function

$$Corr\left[Z(\boldsymbol{c}_i), Z(\boldsymbol{c}_j)\right] = R\left(\boldsymbol{c}_i, \boldsymbol{c}_j\right), \tag{15}$$

where $R\left(\boldsymbol{c}_i, \boldsymbol{c}_j\right)$ denotes the correlation between the random variables $Z(\boldsymbol{c}_i)$ and $Z(\boldsymbol{c}_j)$. Following existing common practices [67], the squared-exponential correlation function is employed:

$$R\left(\boldsymbol{c_i}, \boldsymbol{c_j}\right) = exp\left[-\sum_{k=1}^{d} \theta_k \left(c_{ik} - c_{jk}\right)^2\right], \tag{16}$$

where c_{ik} is the k-th element of $\boldsymbol{c}_i$.

A key advantage of kriging is that a closed-form expression can be obtained for predicting the unknown function $Y(\boldsymbol{c})$ at unobserved locations. If it is supposed the function of interest, Y, is observed at the design settings $\{\boldsymbol{c}_i\}_{i=1}^{n}$, the observation vector becomes $\boldsymbol{y} = [Y(\boldsymbol{c}_1), \ldots, Y(\boldsymbol{c}_n)]^T$. Having observed $\boldsymbol{y}$, [67] the conditional mean of the process at a new point $\boldsymbol{c}_{new}$ is given by

$$\hat{Y}(\boldsymbol{c}_{new}) = \mathbb{E}[Y(\boldsymbol{c}_{new})|\boldsymbol{y}] = \hat{\mu} + \boldsymbol{r}^T \boldsymbol{R}^{-1}(\boldsymbol{y} - \mathbf{1}_n \hat{\mu}), \tag{17}$$

where $\hat{\mu} = \mathbf{1}_n{}^T \boldsymbol{R}^{-1} Y / \mathbf{1}_n{}^T \boldsymbol{R}^{-1} \mathbf{1}_n$ is the estimated value of μ, and $\mathbf{1}_n$ is an n-vector of 1's. Here, $\boldsymbol{R}$ is an $n \times n$ matrix whose (i, j)-th entry is $R(\boldsymbol{c_i}, \boldsymbol{c_j})$, and $\boldsymbol{r}$ is an n-vector whose i_{th} entry is $R(\boldsymbol{c_{new}}, \boldsymbol{c_i})$. A more detailed derivation of Equation (17) can be found in previous works [67, 73] and Appendix 1.

While the predictor in Equation (17) is simple to evaluate when the desired function Y is a scalar function, it becomes much more difficult to evaluate for the problem at hand, where the desired function is spatiotemporal. In particular, there are over 400,000 grid points and 1,000 timesteps for each simulation case, and performing kriging for each grid point and time step would be impractical and time consuming. From a statistical perspective, the use of separate kriging models over each grid point and time step also leads to a serious problem of over-parametrization (as each model requires d correlation parameters), which then results in poor prediction performance for the trained model. Accounting for these challenges and the grid systems remaining static for all simulated cases, an improved kriging-based model is introduced that combines the POD information from each case in the form of a "*weighting number*."

The key idea in KSPOD is to apply the kriging equation, Equation (17), to predict the weight of each POD mode at a new design setting. To this end, the observations of $\boldsymbol{y}$ (i.e., $f(\boldsymbol{x}, t; \boldsymbol{c}_{new})$) are now assumed to be the unit vector $\boldsymbol{e_i}$, where $\boldsymbol{e_i}$ is an n-vector with 1 in its i-th element and 0 elsewhere. Intuitively, this quantifies the fact that the POD information extracted in the i-th design setting corresponds to only that setting and not the other $n-1$ settings. With this in mind, the resulting predictor in Equation (17) can be viewed as the predicted weight for that particular POD term at a new design setting $\boldsymbol{c}_{new}$, which is denoted as $\widehat{w}_{new,i}$. This procedure is repeated for each of the n unit vectors, $(\boldsymbol{e_i})_{i=1}^{n}$, from which the n weighting numbers $\left(\widehat{w}_{new,i}\right)_{i=1}^{n}$ can be obtained. They are subsequently used to predict the new POD modes and coefficients through a weighted average of the extracted modes and coefficients at the new design settings. Once the model

is trained, the predictor is used with the KSPOD expansion to predict the flow evolution at a new design point, that is,

$$\hat{f}(\boldsymbol{x}, t; \boldsymbol{c}_{new}) = \sum_{k=1}^{K} \hat{\beta}_k(\boldsymbol{c}_{new}, t)\hat{\phi}^k(\boldsymbol{c}_{new}, \boldsymbol{x}) \tag{18}$$

and

$$\hat{\phi}^k(\boldsymbol{c}_{new}, \boldsymbol{x}_j) = \frac{\sum_{i=1}^{n} \widehat{w}_i(\boldsymbol{c}_{new})\phi^k(\boldsymbol{c}_i, \boldsymbol{x}_j)}{\sum_{i=1}^{n} \widehat{w}_i(\boldsymbol{c}_{new})} \tag{19}$$

Algorithm 2 outlines the detailed steps in the KSPOD algorithm. First, POD is performed for each simulated geometry to extract the coherent structures. Next, the coefficients of POD modes are trained by ordinary kriging models using the Gaussian kernel in Equation (16) as the correlation function, with the correlation parameter θ tuned using MLE as implemented in the R package "GPfit" [74]. The predictive function can be constructed based on Equation (17). The weighting numbers are also trained using the procedure described above. Lastly, the POD modes and coefficients are predicted, which are used to "reconstruct" the flowfield at the new design setting $\boldsymbol{x}_{new}$.

As with any physical or statistical model, implicit assumptions exist. First, by predicting the i-th POD term of the new design setting using only the information for the i-th POD terms extracted from observed design settings, the ranking of the extracted flow physics from POD (corresponding to the rank for its corresponding expansion term in Equation (13)) is assumed to be invariant over different geometry settings. In other words, the flow feature for the first POD mode corresponds to the same coherent structure over all n simulated design settings. The same holds true for the subsequent modes. Second, for the

methodology to work, design settings whose dominant POD modes capture similar physics should be clustered together for training if they share similar dynamics. As information can be dampened or even canceled during the training process, clustering the data and using cases with similar phase content can mitigate the problem.

Algorithm 2—Kernel-smoothed proper orthogonal decomposition (KSPOD)

DATA:	For each design setting in $\{\boldsymbol{c}_i\}_{i=1}^{n}$, the flow evaluation at each spatial location and time step $f(\boldsymbol{x}_j, t_q; \boldsymbol{c}_i)$ is provided, where $\{\boldsymbol{x}_j\}_{j=1}^{J}$ is the spatial location and $\{t_q\}_{q=1}^{m}$ is the time step.
TRAINING:	*Step 1:* For each design setting $\boldsymbol{x}_i$, proper orthogonal decomposition (POD) is performed and can be written as $f(\boldsymbol{x}_j, t_q; \boldsymbol{c}_i) = \sum_{k=1}^{K} \beta^k(\boldsymbol{c}_i, t_q)\phi^k(\boldsymbol{c}_i, \boldsymbol{x}_j)$.
	Step 2: For each time step t_q and each mode k, an ordinary kriging model is performed on $\{\beta^k(\boldsymbol{c}_1, t_q), \dots, \beta^k(\boldsymbol{c}_n, t_q)\}$ with inputs $\{\boldsymbol{c}_1, \dots, \boldsymbol{c}_n\}$. The predictive function at an untried setting x_{new} is $\hat{\beta}^k(\boldsymbol{c}_{new}, t_q)$.
	Step 3: For$i = 1, \dots, n$, an ordinary kriging model is performed on $\boldsymbol{e}_{\mathrm{i}}$ with inputs $\{\boldsymbol{c}_1, \dots, \boldsymbol{c}_n\}$, and the predictive function at an untried setting $\boldsymbol{x}_{new}$ is $\widehat{w}_i(\boldsymbol{c}_{new})$. Therefore, for each spatial location u_j and each mode k, the predictive function of $\phi^k(\boldsymbol{c}_{new}, \boldsymbol{x}_j)$ is $\hat{\phi}^k(\boldsymbol{c}_{new}, \boldsymbol{x}_j) = \sum_{i=1}^{n} \widehat{w}_i(\boldsymbol{c}_{new})\phi^k(\boldsymbol{c}_i, \boldsymbol{x}_j) / \sum_{i=1}^{n} \widehat{w}_i(\boldsymbol{c}_{new})$.
PREDICTION:	At an untried setting x_{new} is computed as $\hat{\beta}^k(\boldsymbol{c}_{new}, t_q), q = 1, \dots, m; k = 1, \dots, K$ and $\hat{\phi}^k(\boldsymbol{c}_{new}, \boldsymbol{x}_j), j = 1, \dots, J; k = 1, \dots, K.$ Then, $f(\boldsymbol{x}_j, t_q; \boldsymbol{c}_{new}) = \sum_{k=1}^{K} \hat{\beta}^k(\boldsymbol{c}_{new}, t_q)\hat{\phi}^k(\boldsymbol{c}_{new}, \boldsymbol{x}_j)$, where $j = 1, \dots, J$ and $q = 1, \dots, m$.

This method is termed KSPOD, because the kriging here does not apply the weighting number $\widehat{w}_{new,i}$ to the flowfield directly. The vortex structures within the flowfield are a combination of waves with different frequencies, amplitudes, and phases. If the weighting number is used on the flowfield directly, the two datasets may cancel each other out during the regression process, thereby eliminating useful information. The phase difference can be observed in POD modes as well. Application of weighting functions to POD modes with a kernel-smoothed algorithm can prevent the phase cancelation and retain important flow physics. Once the emulator model is trained, it can be used with the KSPOD model for predicting the flow evolution at a new design point. The computation cost is reduced by several orders of magnitude. The high-fidelity original simulations take around 107,000 CPU hours for each case. The trained model can evaluate a new flowfield in 0.02 CPU hours.

3.3.2.1 Interpretation as a Nadaraya-Watson Kernel Smoother

It is supposed that simulations are to be conducted at various design geometries $\boldsymbol{c} = \{\boldsymbol{c_1}, \dots, \boldsymbol{c_n}\}$, and it is assumed that the true function $W(\boldsymbol{c})$ is a realization from a stochastic process

$$W(\boldsymbol{c}) = \mu + Z(\boldsymbol{c}), \tag{20}$$

where $\boldsymbol{c}$ is an n-dimentional vector (with d design variables), μ a constant global model, and $Z(\boldsymbol{c})$ is a local deviation from the global model with zero mean. It is considered that the kriging has the indicator vector $\boldsymbol{e}_i$, an n-vector with unity in the i-th entry, and zero elsewhere. There is another vector $\boldsymbol{r_i}$, where $i = 1, \cdots, p$, and p is the number of control

settings for design cases. Since a space-filling design is employed, it is simple to show that the optimal correlation parameters for the underlying GP should be equal for all p dimensions. This common correlation is denoted as θ. When the number of design points $n \to \infty$, one can show that $\theta \to \infty$ as well, since the "kriging surface" for $\boldsymbol{e}_i$ converges pointwise to a discontinuous surface with value 1 at $\boldsymbol{r}_i$ and 0 elsewhere. The kriging estimate, $\boldsymbol{e}_i$, for a new design setting, $\boldsymbol{r}_{new}$, is

$$\widehat{w}_{new,i} = \hat{\mu} + \boldsymbol{r}_{new}^T \boldsymbol{R}^{-1}(\boldsymbol{e}_i - \hat{\mu}\mathbf{1}_n), \tag{21}$$

where $\hat{\mu} = (\mathbf{1}_n^T \boldsymbol{R}^{-1}\mathbf{1}_n)^{-1}\mathbf{1}_n^T \boldsymbol{R}^{-1}$, $\boldsymbol{e}_i = \boldsymbol{R}_{ii}^{-1} / \sum_{i=1}^n \boldsymbol{R}_{ii}^{-1}$, $\boldsymbol{R} \equiv Corr\left[Z(\boldsymbol{c_i}), Z(\boldsymbol{c_j})\right]$ are $n \times n$ correlation matrices, and $\widehat{w}_{new,i}$ is a weighted number based on kriging. When $\theta \to \infty$ and $n \to \infty$, the best linear unbiased predictor estimator $\hat{\mu} \to 0$; when $\theta \to \infty$, the inverse correlation matrix $\boldsymbol{R}^{-1}$ converges element-wise to $\mathbf{I}_n$. Under these two approximations, a new kernel is created:

$$k_\theta(\boldsymbol{c}_i, \boldsymbol{c}_{new}) = \widehat{w}_{new,i} \approx \boldsymbol{r}_{new}^T \boldsymbol{R}^{-1}\boldsymbol{e}_i = \exp\left\{-\theta\left|\left|\boldsymbol{c}_i - \boldsymbol{c}_{new}\right|\right|_2^2\right\}, \tag{22}$$

where $||\cdot||_2$ is the Euclidean norm. In other words, $\widehat{w}_{new,i}$ is the isotropic Gaussian kernel $k_\theta(\boldsymbol{c}_i, \boldsymbol{c}_{new})$. The proposed predictor of the first k-th mode at the new design setting $\boldsymbol{c}_{new}$ is

$$\hat{\phi}_{new}^k(\boldsymbol{x}) = \frac{\sum_{i=1}^n \widehat{w}_{new,i}\phi_i^k(\boldsymbol{x})}{\sum_{i=1}^n \widehat{w}_{new,i}} \approx \frac{\sum_{i=1}^n k_\theta(\boldsymbol{c}_i, \boldsymbol{c}_{new})\phi_i^k(\boldsymbol{x})}{\sum_{i=1}^n k_\theta(\boldsymbol{c}_i, \boldsymbol{c}_{new})} \tag{23}$$

where $\phi_i^k(\boldsymbol{x}), i = 1, \cdots, n$ for the k-th POD mode at design setting $\boldsymbol{c}_i$.

With kriging over k modes, the new POD coefficient is defined as

$$\hat{\beta}_{new}^{k}(\boldsymbol{x},t) = \left(\hat{\beta}_{1}^{k},\hat{\beta}_{2}^{k},\ldots,\hat{\beta}_{10d}^{k}\right), \tag{24}$$

and the corresponding POD expansion using k_{th} modes (that is, the prediction) is given by

$$\hat{f}_{new}(\boldsymbol{x},t) = \sum_{k=1}^{\infty}\hat{\beta}_{new}^{k}(\boldsymbol{x},t)\hat{\phi}_{new}^{k}(\boldsymbol{x}) = \sum_{k=1}^{\infty}\frac{\beta_{i}^{k}(\boldsymbol{x},t)\left(\sum_{i=1}^{n}k_{\theta}(\boldsymbol{c}_{i},\boldsymbol{c}_{new})\phi_{i}^{k}(\boldsymbol{x})\right)}{\left(\sum_{i=1}^{n}k_{\theta}(\boldsymbol{c}_{i},\boldsymbol{c}_{new})\right)^{2}} \tag{25}$$

Equation (22) can be viewed as a kernel smoother on the observed modes $\left\{\phi_{i}^{k}(\boldsymbol{x})\right\}_{i=1}^{n}$ and coefficients $\left\{\beta_{i}^{k}(\boldsymbol{x},t)\right\}_{i=1}^{n}$.

3.3.3 Common Kernel-Smoothed Proper Orthogonal Decomposition (CKSPOD)

This section introduces the novel idea of CKSPOD, which combines statistical modeling with data-reduction methods to improve the accuracy of a surrogate emulation model. The key advantage of CKSPOD is that it not only provides an efficient method for training a reduced-basis model using simulated flows at observed design settings but also that it allows for flow predictions over the desired design space in practical turnaround times. Moreover, it fixes the phase differences (i.e., sign differences of eigen-vectors) and time-delay issues found in KSPOD by adding a transfer matrix in a singular-value decomposition (SVD) step to the data-reduction process.

A classic POD, as introduced in Section 0, can be written in the following form:

$$f(\boldsymbol{x},t) \approx \sum_{k=1}^{K} \beta^k(t)\phi^k(\boldsymbol{x}), \tag{13}$$

where $f(\boldsymbol{x},t)$ is the simulated flowfield at spatial location $\boldsymbol{x}$ and time t, and $\beta^k(t)$ and $\phi^k(\boldsymbol{x})$ represent the time-varying coefficient and basis function for the i-th mode, respectively. As indicated in Equation (13), the expansion is truncated with the first K terms, where K is chosen such that the reconstructed flowfield retains a desired degree of accuracy. In practice, the time-varying coefficients and basis functions are obtained through an eigen-decomposition of the inner product of a flowfield variable [15]. Equation (13) can be viewed as the optimal decomposition of $f(\boldsymbol{x},t)$, using a basis expansion of K terms.

As mentioned in Chang et al. [3], from a physics perspective, POD provides valuable insight into the key physics present in the flowfield. The basis function, or mode shape $\phi^k(\boldsymbol{x})$, can be interpreted as the spatial distribution of the fluctuation field of a given flow variable (such as pressure, density, temperature, or velocity components). It represents the dominant coherent structure, such as acoustic waves in the system [24]. A spectral analysis of the POD coefficients, $\beta^k(t)$, carrying time information, can be performed to identify flow periodicity and characteristic frequencies for hydrodynamic and acoustic instabilities. The index for the basis function expansion in Equation (13) is determined by the rank of the energy content in the eigen-decomposition calculation and suggests which flow structure is more prevalent. The first few terms in the expansion represent more energy-containing structures, and the remaining terms represent increasingly weaker flow features.

In KSPOD work, flow features are extracted using POD within a statistical framework, allowing the training of an emulator for flow prediction. The surrogate model utilizes the machine-learning technique GPR, also known as kriging, to predict POD modes and time-varying coefficients at a new design setting.

A key advantage of kriging is that a closed-form expression can be obtained for predicting the unknown function $Y(\boldsymbol{c})$ at unobserved locations. It if it supposed that the function of interest Y is observed at the design setting $\{\boldsymbol{c}_i\}_{i=1}^{n}$, the observation vector then becomes $\boldsymbol{y} = [Y(\boldsymbol{c}_1), \dots, Y(\boldsymbol{c}_n)]^T$. Having observed $\boldsymbol{y}$, the mathematical framework for kriging of the prediction value $\hat{Y}$ at a new point $\boldsymbol{c}_{new}$ is given by

$$\hat{Y}(\boldsymbol{c}_{new}) = \mathbb{E}[Y(\boldsymbol{c}_{new})|\boldsymbol{y}] = \hat{\mu} + \boldsymbol{r}^T\boldsymbol{R}^{-1}(\boldsymbol{y} - \mathbf{1}_n\hat{\mu}), \tag{17}$$

where $\hat{\mu} = \mathbf{1}_n{}^T\boldsymbol{R}^{-1}Y/\mathbf{1}_n{}^T\boldsymbol{R}^{-1}\mathbf{1}_n$ is the estimated value of μ, and $\mathbf{1}_n$ is an n-vector of 1's. Here, $\boldsymbol{R}$ is an $n \times n$ matrix of squared-exponential correlation function whose (i,j)-th entry is $R(\boldsymbol{c}_i, \boldsymbol{c}_j)$, and $\boldsymbol{r}$ is an n-vector whose i_{th} entry is $R(\boldsymbol{c}_{new}, \boldsymbol{c}_i)$. A more detailed derivation of Equation (17) can be found in a number of previous works [67, 71-73] and Appendix 1.

It is extremely time consuming to predict a new design with spatial point-to-point kriging along the timesteps. The key idea in KSPOD is to apply the kriging equation, Equation (17), to predict the weight of each POD mode at a new design setting. For KSPOD, POD is first applied for each simulated case to extract coherent structures. Next, the coefficients of POD modes are trained by ordinary kriging models using Equation (17). The weighting numbers are also trained based on the relationship between the new design

point and each given number. The POD modes of new points are calculated by multiplying the POD modes of the training cases by the weighting number. Finally, the prediction spatiotemporal evolving flow dynamics at the new design setting $\boldsymbol{c}_{new}$ are reconstructed by the new modes and coefficients.

To calculate the weighting number of the POD modes for each training case, the observations $\boldsymbol{y}$ are now taken to be the unit vector $\boldsymbol{e_i}$, where $\boldsymbol{e_i}$ is an n-vector with 1 in its i-th element and 0 elsewhere. Intuitively, this quantifies the fact that the POD information extracted in the i-th design setting corresponds to only that setting and not the other n-1 settings. With this in mind, the resulting predictor in Equation (17) can be viewed as the predicted weight for that particular POD term at a new design setting $\boldsymbol{c}_{new}$, denoted as $\widehat{w}_{new,i}$. This procedure is repeated for each of the n unit vectors $(\boldsymbol{e_i})_{i=1}^{n}$, from which the n weighting numbers $\left(\widehat{w}_{new,i}\right)_{i=1}^{n}$ can be obtained. They are subsequently used to predict the new POD modes and coefficients through a weighted average of the extracted modes and coefficients at the new design settings.

Although KSPOD can successfully predict spatiotemporal flowfields in a simplex swirl injector, the details of flow dynamics between a simulation and its prediction are not always similar. Time delays of evolving flow structures are sometimes demonstrated, because KSPOD performs POD on training cases individually. Therefore, the signs of eigen-vectors of these training cases may not be the same. This sign difference can cause 180-degree phase differences in POD modes between any two cases.

Similar problems do not occur in CPOD, in which all training information is concatenated into a gigantic training matrix to create the CPOD covariance matrix.

Therefore, CPOD only performs POD once and applies kriging to predict POD coefficients. Subsequently, all dominant coherent structures from training cases are combined. In other words, the set of POD modes derived by CPOD can only preserve spatial physics information for the mean flowfield. Moreover, the prediction is derived from the same set of POD modes.

To improve this surrogate model, advantages from the two methods described above must be incorporated and disadvantages must be excluded. The new method must first performs POD for all training cases individually to preserve the dominate coherent structures of each design. Also, the new model must operate eigen-decomposition for all cases simultaneously to make ensure that opposite signs between their eigen-vectors do not exist. As such, a transfer matrix must be employed during SVD for all cases before performing POD.

The flowfield information can be written in the form of SVD as the following:

$$f(\boldsymbol{x}, t) = A = \mathrm{USV}^{\mathrm{T}}\,, \tag{26}$$

where $A \in R^{n\times m}$, U is a $n \times k$ orthonormal matrix spanning A's column space $\mathrm{im}(A)$, S is a $k \times k$ diagonal matrix of singular values, and V is $m \times k$ orthonormal matrix spanning A's row space $\mathrm{im}(A^T)$. Here, k represents the first truncated k modes where the underlying flow is approximated to a desired degree of accuracy. The covariance, POD modes, and POD coefficients of A can be written as

$$cov(A, A) = C = \frac{A^T A}{n-1} = \frac{(USV^T)^T USV^T}{n-1} = \frac{VS^T U^T USV^T}{n-1} = \frac{VS^2 V^T}{n-1} = \frac{VLV^T}{n-1}\,, \tag{27}$$

$$\phi^k(\boldsymbol{x}) = AV\ , \tag{28}$$

$$\beta^k(t) = \frac{(\phi^k)^T A}{L} = \frac{(AV)^T A}{L} = \frac{V^T A^T A}{L} = V^T \frac{VLV^T}{L(n-1)}(n-1) = V^T\ . \tag{29}$$

To maintain $\phi^k(\boldsymbol{x})$ and $\beta^k(t)$ for all training cases with same phase (or sign), the SVD processes of these training cases must occur simultaneously without affecting the definition of POD. To satisfy these conditions, the SVD can be modified by the Hadamard product, such as

$$\begin{aligned} \mathbb{C} = C_1 \circ C_2 \circ \ldots \circ C_H &= \left(\frac{V_1 L_1 V_1^T}{n-1}\right) \circ \ldots \circ \left(\frac{V_H L_H V_H^T}{n-1}\right) \\ &= \left(\frac{1}{n-1}\right)^H (V_1 L_1 V_1^T) \circ \ldots \circ (V_H L_H V_H^T) = \mathbb{V}\mathbb{L}\mathbb{V}^T, \end{aligned} \tag{30}$$

where C_i represents the covariance matrix for case i, $i = 1,2,\ldots,H$, and H represents the total number of design cases. Because all covariance matrices are positive semidefinite, they can be represented as $\mathrm{C}_i = M_i M_i^T$. Therefore, Equation (30) can be re-written as

$$\mathbb{C} = C_1 \circ C_2 \circ \ldots \circ C_H = \mathbb{V}\mathbb{L}\mathbb{V}^T = (M_1 M_1^T) \circ \ldots \circ (M_H M_H^T). \tag{31}$$

Based on the characteristics of the Hadamard product, the CKSPOD covariance $\mathbb{C}$ can be organized as

$$\begin{aligned} \mathbb{C} &= (M_1 M_1^T) \circ \ldots \circ (M_H M_H^T) \\ &= (M_1 M_1^T) \circ (\Sigma(M_2 \circ M_3 \circ \ldots \circ M_H)(M_2 \circ M_3 \circ \ldots \circ M_H)^T) \end{aligned} \tag{32}$$

$$= \frac{V_1 L_1 V_1^T}{n-1} \circ (\Sigma (M_2 \circ M_3 \circ \dots \circ M_H)(M_2 \circ M_3 \circ \dots \circ M_H)^T)$$

$$= (M_2 M_2^T) \circ (M_1 M_1^T) \circ (M_3 M_3^T) \circ \dots \circ (M \quad M_H^T)$$

$$= \frac{V_2 L_2 V_2^T}{n-1} \circ (\Sigma (M_1 \circ M_3 \circ \dots \circ M_H)(M_1 \circ M_3 \circ \dots \circ M_H)^T)$$

$$= \frac{V_i L_i V_i^T}{n-1} \circ \left(\sum_{\substack{j=1 \\ j \neq i}}^{H} (M_j \circ \dots \circ M_H)(M_j \circ \dots \circ M_H)^T \right)$$

$$= \mathbb{V}\mathbb{L}\mathbb{V}^T \quad ,$$

where the new POD modes and coefficients can be represented as

$$\tilde{\phi}^k(\boldsymbol{c}) = A_i \mathbb{V}$$

$$= A_i \frac{V_i L_i V_i^T}{n-1} \circ \left(\sum_{\substack{j=1 \\ j \neq i}}^{H} (M_j \circ \dots \circ M_H)(M_j \circ \dots \circ M_H)^T \right) \mathbb{V}\mathbb{L}^{-1}$$

$$= A_i V_i \frac{L_i V_i^T}{n-1} \circ \left(\sum_{\substack{j=1 \\ j \neq i}}^{H} (M_j \circ \dots \circ M_H)(M_j \circ \dots \circ M_H)^T \right) \mathbb{V}\mathbb{L}^{-1}$$

$$\cong A_i V_i \mathcal{T}$$

$$= \phi^k(\boldsymbol{x}) \mathcal{T} \, . \tag{33}$$

Therefore,

$$\mathcal{T} = \frac{L_i V_i^T}{n-1} \circ \left(\sum_{\substack{j=1 \\ j \neq i}}^{H} (M_j \circ \dots \circ M_H)(M_j \circ \dots \circ M_H)^T \right) \mathbb{V}\mathbb{L}^{-1} \tag{34}$$

can be viewed as a CKSPOD transfer matrix, the new POD modes can be written as

$$\widetilde{\Phi}^k(\boldsymbol{c}) = \frac{\widetilde{\phi}^k(\boldsymbol{c})}{norm(\widetilde{\phi}^k(\boldsymbol{c}))} \quad , \tag{35}$$

$$\tilde{\beta}^k(t) = \mathbb{V}^T norm(\widetilde{\Phi}^k(\boldsymbol{c})) \quad , \tag{36}$$

and the CKSPOD expansion to predict the flow evolution at a new design point is,

$$\hat{f}(\boldsymbol{x}, t; \boldsymbol{c}_{new}) = \sum_{k=1}^{K} \tilde{\beta}^k(t)(\boldsymbol{c}_{new}, t) \frac{\sum_{i=1}^{n} \widehat{w}_i(\boldsymbol{c}_{new}) \widetilde{\Phi}^k(\boldsymbol{c}_i, x_j)}{\sum_{i=1}^{n} \widehat{w}_i(\boldsymbol{c}_{new})} \tag{37}$$

Algorithm 3 outlines the detailed steps in the CKSPOD algorithm. First, POD is performed for each simulated flow to extract the coherent structures (Step 1 of the training process in Algorithm 3). Next, in Step 2 of training process, the coefficients of the POD modes are trained by ordinary kriging models using the Gaussian kernel in Equation (17) as a correlation function, with correlation parameter θ trained with MLE, using the R package "GPfit" [74]. The predictive function can be constructed based on Equation (17). In Step 3 of training process, the weighting numbers are trained using the procedure described above. Note that the estimated correlation parameter θs of Steps 2 and 3 are different. Lastly, in the prediction process, the POD modes and coefficients are predicted at the new design setting $\boldsymbol{c}_{new}$, which yields the flow field at setting $\boldsymbol{c}_{new}$ by reconstruction.

As with any physical or statistical model, there are several implicit assumptions for the proposed method. First, by predicting the i-th POD term of the new design setting using only the information for the i-th POD terms extracted from observed design settings, the ranking of the extracted flow physics from POD (corresponding to the rank for its

corresponding expansion term in Equation (13)) is assumed to be invariant over different geometry settings. In other words, the flow instability for the first POD mode corresponds to the same coherent structure over all n simulated design settings, and the same holds for the remaining modes. Second, for the methodology to work, design settings whose dominant POD modes capture similar physics should be clustered together for training if they share similar wave information. This is partially because in the weighting prediction from a kriging predictor, the wave information from the flow can be dampened or canceled in the prediction process, and a clustering of the data beforehand can mitigate this problem by training cases with similar wave information.

Because all cases sharing the common eigen-decomposition during data reduction via Hadamard product, the method is termed as common kernel-smoothed POD technique than kriging, especially if the latter leverages different correlation parameters over each dimension to approximate an unknown surface. Here, kriging would not apply the weighting number, $\widehat{w}_{new,i}$, to the flow field directly, since the unstable flowfield combines waves with different frequencies, amplitudes, and phases. Simulation cannot control the starting phase for each wave. If the weighting number is used for the flowfield directly, it is possible that the two training data sets cancel each other during kriging, eliminating some flow information. The phase difference can also be observed in POD modes. Applying the weighting function to POD modes with kernel-smoothed algorithms can avoid the phase neutralization and retain important physics information.

Once the emulator model is trained, it can be used with the KSPOD model for predicting the flow evolution at a new design point. The computation cost is reduced by a few orders of magnitudes. Simulation data typically take 10-14 days, or around 107,136

CPU hours, on average. After training the model with sufficient samples, each snapshot of flowfields with new design can be evaluated within a few seconds.

Algorithm 3—Common kernel-smoothed proper orthogonal decomposition (CKSPOD)

DATA:	For each design setting in $\{\boldsymbol{x}_i\}_{i=1}^{H}$, the flow evaluation at each spatial location and time step $f(\boldsymbol{x}_j, t_q; \boldsymbol{c}_i)$ is provided, where $\{\boldsymbol{x}_j\}_{j=1}^{J}$ is the spatial location and $\{t_q\}_{q=1}^{m}$ is the time step.	
TRAINING:	*Step 1:*	The Hadamard product is calculated with covariance matrices from each design setting $\boldsymbol{c}_i$, $\mathbb{C} = \mathrm{cov}(f(\boldsymbol{c}_1), f(\boldsymbol{c}_1)) \circ \mathrm{cov}(f(\boldsymbol{c}_2), f(\boldsymbol{c}_2)) \circ \ldots \circ \mathrm{cov}(f(\boldsymbol{c}_H), f(\boldsymbol{c}_H)) = \mathrm{C}_1 \circ \mathrm{C}_2 \circ \ldots \circ \mathrm{C}_H$. Proper orthogonal decomposition (POD) based on singular-value decomposition (SVD) with $\mathbb{C}$ is performed and can be written as $f(\boldsymbol{x}_j, t_q; \boldsymbol{c}_i) = \sum_{k=1}^{K} \hat{\beta}^k(\boldsymbol{c}_i, t_q)\tilde{\phi}^k(\boldsymbol{c}_i, \boldsymbol{x_j})$. Note that: 1) $\tilde{\Phi}^k(\boldsymbol{c}_i, \boldsymbol{x}_j) = \frac{\tilde{\phi}^k(\boldsymbol{x})}{norm(\tilde{\phi}^k(\boldsymbol{x}))}$ and $\tilde{\boldsymbol{\phi}}^k(\boldsymbol{x}) = A_i\mathbb{V}$; 2) $\tilde{\beta}^k(\boldsymbol{c}_i, t_q) = \mathbb{V}^{\mathrm{T}} norm\left(\hat{\phi}^k(\boldsymbol{c}_i, \boldsymbol{x}_j)\right)$.
	Step 2:	For each time step t_q and each mode k, an ordinary kriging model is performed on $\{\tilde{\beta}^k(\boldsymbol{c}_1, t_q), \ldots, \tilde{\beta}^k(\boldsymbol{c}_H, t_q)\}$ with inputs $\{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_H\}$, and the predictive function at an untried setting x_{new} is $\hat{\beta}^k(\boldsymbol{c}_{new}, t_q)$.
	Step 3:	For $i = 1, \ldots, H$, an ordinary kriging model is performed on $\boldsymbol{e}_{\mathrm{i}}$ with inputs $\{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_H\}$, and the predictive function at an untried setting $\boldsymbol{c}_{new}$ is $\widehat{w}_i(\boldsymbol{c}_{new})$. Therefore, for each spatial location u_j and each mode k, the predictive function of $\phi^k(\boldsymbol{c}_{new}, x_j)$ is $\hat{\phi}^k(\boldsymbol{c}_{new}, x_j) = \frac{\sum_{i=1}^{n} \widehat{w}_i(\boldsymbol{c}_{new})\tilde{\Phi}^k(\boldsymbol{c}_i, x_j)}{\sum_{i=1}^{n} \widehat{w}_i(\boldsymbol{c}_{new})}$.
PREDICTION:	At an untried setting x_{new}, it is computed that $\hat{\beta}^k(\boldsymbol{c}_{new}, t\ \ ), q = 1, \ldots, m; k = 1, \ldots, K$, and $\hat{\phi}^k(\boldsymbol{c}_{new}, x_j), j = 1, \ldots, J; k = 1, \ldots, K$, then $\hat{f}(x_j, t_q; \boldsymbol{c}_{new}) = \sum_{k=1}^{K} \hat{\beta}^k(\boldsymbol{c}_{new}, t_q)\hat{\phi}^k(\boldsymbol{c}_{new}, x_j)$, where $j = 1, \ldots, J$ and $q = 1, \ldots, m$.	

3.4 Data-Driven Framework for Emulation Analysis

3.4.1 Sensitivity Analysis

The first component of the data-driven framework for emulation analysis is a sensitivity analysis using Sobol indices [75] to identify which design parameters contribute more to changes in responses of interest, such as liquid-film thickness or spreading angle. This analysis is also valuable for parameter reduction. This method aims to decompose the variations of certain desired output variables into the partial variations attributable to each input parameter and the effects of interactions between parameters. Such a method of analyzing sensitivity connects closely to the classical analysis of variance employed in linear regression models [76].

To state this concept in mathematical terms, $f(\boldsymbol{c})$ is the desired response output at design setting $\boldsymbol{c}$, where $\boldsymbol{c} = (c_1, c_2, \dots, c_p)$ corresponds to the input parameters over a unit hypercube $[0,1]^p$. Specifically, for the current study, $p = 5$, $c_1 = L, c_2 = R_n, c_3 = \theta, c_4 = \delta$, and $c_5 = \Delta L$, with the design range for all parameters normalized to the interval $[0,1]$. The random variable $\boldsymbol{X}$ is defined as a uniform distribution over $[0,1]^p$, $f_0 = \mathbb{E}[f(\boldsymbol{X})]$ is the response mean, and $D = \mathrm{Var}[f(\boldsymbol{X})]$ is the response variance over the design range. The goal is to decompose the response variance, D, into the contributions for each design parameter, $c_1, \cdots, c_p$, as well as the effects of interactions between parameters. Consider the following decomposition,

$$f(\boldsymbol{c}) = f_0 + \sum_{i=1}^{p} f_i(c_i) + \sum_{1 \le i < j \le p} f_{i,j}(c_i, c_j) + \cdots + f_{1,2,\cdots,p}(c_1, \cdots, c_p), \tag{38}$$

where each summand satisfies

$$\int_0^1 f_{i_1,\dots,i_t}(c_{i_1},\cdots,c_{i_t})\,dc_k = 0 \tag{39}$$

for any $k = i_1,\dots,i_t$ and has orthogonal components. In Equation (38), the main effect index of input i is

$$f_i(c_i) = \int (f(c) - f_0)\,dc_{-i}, \quad c_{-i} = \{c_1,\cdots,c_p\}\backslash\{c_i\} \tag{40}$$

and the two-way interaction index of inputs i and j is

$$\begin{gathered} f_{i,j}(c_i,c_j) = \int \{f(c) - f_0 - f_i(c_i) - f_j(c_j)\}\,dc_{-i,j}, \\ c_{-i,j} = \{c_1,\cdots,c_p\}\backslash\{c_i,c_j\}. \end{gathered} \tag{41}$$

Squaring both sides of Equation (38) and taking the integral over $[0,1]^p$, the result is

$$D = \sum_{i=1}^{p} D_i + \sum_{1\le i<j\le p} D_{ij} + \sum_{1\le i<j<l\le p} D_{ijl} + \cdots + D_{1,2,\dots,p}, \tag{42}$$

where D_u is the partial variance corresponding to a subset of parameters $u \subseteq \{1,\cdots,p\}$:

$$D_u = \int f_u^2(c_u)dc_u. \tag{43}$$

The Sobol sensitivity indices for parameter subset u can be defined as [75]

$$S_u = \frac{D_u}{D} \in [0,1], \tag{44}$$

with larger values of S_u indicating greater importance of the interaction effect for u.

In practice, Sobol indices can be estimated as follows. First, a pseudo-random parameter sequence is generated using a low discrepancy Sobol point set [77]. Second, this sequence is used to approximate the above integrals, which can then provide estimates for the corresponding Sobol indices. The quantification of the response sensitivity for each parameter serves two purposes: (1) it provides a preliminary analysis of important effects in the system, which can guide further physical investigations, and (2) it allows for a reduction in the number of parameters that must be considered in the emulator, thereby providing a computationally efficient method to survey flow properties within the design space. A detailed discussion of the sensitivity analysis is presented in Section 4.2.1 for the current physical model.

3.4.2 Decision Tree

As mentioned in Section 4.1.1, a jet-like/swirling flow dichotomy exists within the design space. For simulated design points, it is simple to classify whether such a parameter combination results in a jet-like or a swirling flow, since the flowfield data are readily available. For design settings that have not been simulated, a data-driven technique is needed to make such a classification. With this technique, a boundary between jet-like and swirling cases can first be established over the design space of interest, which can then be used to gain physical insight into the design space and to guide additional experiments. Second, the classification information can be used to train separate surrogate models within the jet-like and swirling domains. This partitioning of the emulator training dataset allows the model to extract different flow characteristics associated with jet-like and swirling

behavior separately and can thereby improve its predictive accuracy. The powerful machine-learning tool, “decision tree,” is employed for the classification process.

A decision tree is a decision support tool that models decisions and their possible consequences; decision trees are one of the most popular predictive models in data mining and machine learning [78]. Such methods belong to a larger class of learning methods called supervised learning [79], which aims to predict an objective function from labeled training data. A classification tree, a special type of decision tree, is used here. It specializes in predicting classification outcomes, such as whether a parameter set has a jet-like or swirling flow. The trained model can be summarized by a binary tree, separating the design space into two subgroups. Each node of this tree represents a parameter decision, and each leaf of the tree indicates the class of outcomes, following the chain of decisions made from the tree root.

A classification tree can be trained using the following algorithm (see [79] for details). First, the simulated flowfields of each sampled design point are examined and classified as either jet-like or swirling flows, depending on the radial penetration of the propellant in the downstream region. Next, a search is conducted across all design parameters and possible split-points, identifying the parameter constraints that minimize misclassification. A branch is then created in the classification tree corresponding to the parameter constraint. The same branching procedure is repeated for each of the resulting child nodes. For the analysis in Section 4, the Gini impurity index [80] is selected as the misclassification measure

$$\hat{p}_j\left(1 - \hat{p}_j\right) + \hat{p}_s(1 - \hat{p}_s), \tag{45}$$

where $\hat{p}_j$ and $\hat{p}_s$ are the proportions of jet-like and swirl cases in a split. This index measures how often a randomly chosen sample is incorrectly labeled when such a label is randomly assigned from the dataset. Notice that a Gini index of 0 indicates that (a) $\hat{p}_j = 1$ and $\hat{p}_s = 0$, or (b) $\hat{p}_j = 0$ and $\hat{p}_s = 1$, both of which suggest perfect classification. When the Gini impurity index nears 0.5, jet-like and swirl cases are equally distributed. If more than two groups are considered in the injector dynamics, the Gini impurity index can be generalized for other number of groups, which can be seen in Breiman (1984) [80].

This decision-tree learning technique not only provides a means for partitioning the training dataset for the model into jet-like and swirling flows but also reveals physical insights into the important design parameter constraints causing the jet-swirl dichotomy. The quantification of this split is achieved through the calculation of the Gini impurity. When building the decision tree, the Gini index is a criterion to minimize for classification, where 0.5 is the worst classification possible; however, the optimization procedure aims to find the best classification possible (i.e., one with smallest Gini index). If this optimal classification with two categories is not sufficient, the approach should be generalized for classification trees with more than two categories. The interpretability of these constraints is elaborated in Section 3.4.3.1.

3.4.3 Uncertainty Quantification (UQ) for Spatiotemporal Emulation

This section outlines a proposed model for UQ analysis for spatiotemporal emulation models; it first addresses model specifications and parameter estimations for CPOD [1] before extending the application to KSPOD- and CKSPOD-based emulation models.

3.4.3.1 Common Proper Orthogonal Decomposition (CPOD) Model Specifications

In the CPOD model, after the CPOD extraction, the extracted time-varying coefficients $\{\beta_k(t, \boldsymbol{c}_i)\}_{i,k}$ are used as data for fitting the proposed emulator. Some previous work has examined dynamic emulator models [81-83], but the sheer number of simulation timesteps in the current work can impose high computation times and numerical instabilities for these existing methods [84]. As mentioned previously, computational efficiency is paramount for the problem being studied, since simulation runs can be performed within a week. Moreover, existing emulators cannot account for cross-correlations between different dynamic systems, while the flow physics represented by different CPOD modes are known to be highly coupled from governing equations. The dense temporal resolution of the flow is exploited by using a time-independent emulator that employs independent kriging models at each slice of time. The rationale is that, because time scales are so fine, no practical need exists to estimate temporal correlations (even when they exist), since prediction is not required between timesteps. This time-independent simplification is key for emulator efficiency, since it allows for the full exploitation of the power of parallel computing for model fitting and flow prediction.

The model is as follows. It is supposed that $\mathcal{R}$ flow variables are considered (with $\mathcal{R} = 6$ in the present case), and the CPOD expansion in Equation (8) is truncated at $\mathcal{K}_r$ terms for flow $r = 1, \dots, \mathcal{R}$. The equation $\beta^r(t; \boldsymbol{c}) = \left(\beta_1^r(t; \boldsymbol{c}), \dots, \beta_{\mathcal{K}_r}^r(t; \boldsymbol{c})\right)^{\mathrm{T}}$ is the vector of $\mathcal{K}_r$ time-varying coefficients for flow variable r at design setting $\boldsymbol{c}$, with $\beta(t; \boldsymbol{c}) = \left(\beta_1^{(1)}(t; \boldsymbol{c}), \dots, \beta_{\mathcal{K}_r}^{(\mathcal{R})}(t; \boldsymbol{c})\right)^{\mathrm{T}}$ as the coefficient vector for all flows at $\boldsymbol{c}$. The following time-independent GP model on $\beta(t; \boldsymbol{c})$ is assumed:

$$\beta(t;\boldsymbol{c}) \sim GP\{\boldsymbol{\mu}(t), \boldsymbol{\Sigma}(\cdot\,,\cdot\,;t)\}, \beta(t;\boldsymbol{c}) \perp \beta(t';\boldsymbol{c}) \text{ for } t \neq t'. \tag{46}$$

Here, $\mathcal{K} = \sum_{r=1}^{\mathcal{R}} \mathcal{K}_r$ is the number of extracted modes over all R flow variables, $\boldsymbol{\mu} \in \mathbb{R}^{\boldsymbol{\mathcal{K}}}$ is the process mean vector, and $\Sigma(\cdot,\cdot) : \mathbb{R}^{\boldsymbol{p}} \times \mathbb{R}^{\boldsymbol{p}} \to \mathbb{R}^{\boldsymbol{\mathcal{K}}\times\boldsymbol{\mathcal{K}}}$ is its corresponding covariance matrix function, as defined below. Since the GPs are now time-independent, the specification for *fixed* time t is presented, and $\beta(t;\boldsymbol{c})$, $\boldsymbol{\mu}(t)$ and $\boldsymbol{\Sigma}(\cdot\,,\cdot\,;t)$ are referred to as $\beta(\boldsymbol{c})$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}(\cdot\,,\cdot)$ for brevity.

For computational efficiency, the following separable form is assumed for $\boldsymbol{\Sigma}(\cdot\,,\cdot)$:

$$\boldsymbol{\Sigma}(\mathbf{c_1}, \boldsymbol{c_2}) = r_\tau(\boldsymbol{c_1}, \boldsymbol{c_2})\mathbf{T},\ r_\tau(\boldsymbol{c_1}, \boldsymbol{c_2}) = \prod_{j=1}^{p} \tau_j^4 \left(c_{1j} - c_{2j}\right)^2,$$
$$\boldsymbol{c_1}, \boldsymbol{c_2} \in \mathbb{R}^{\boldsymbol{p}}, \tau_j \in (0,1), \tag{47}$$

where $\mathbf{T} \in \mathbb{R}^{\boldsymbol{\mathcal{K}}\times\boldsymbol{\mathcal{K}}}$ is a symmetric, positive definite matrix called the CPOD covariance matrix, and $\tau_r(\cdot\,,\cdot\,)$ is the correlation function over the design space, parameterized by $\boldsymbol{\tau} = \left(\tau_1, \ldots, \tau_p\right)^T \in (0,1)^p$. This can be viewed as a large co-kriging model [85] over the design space, with the multivariate observations being the extracted CPOD coefficients for all flow variables. Note that r_τ is a reparameterization of the squared-exponential (or Gaussian) correlation function $\exp\{-\sum_{j=1}^{p} \theta_j \left(c_{1j} - c_{2j}\right)^2\}$, with $\theta_j = -4 \log \tau_j$. Such a reparameterization allows for a more numerically stable optimization of MLEs, because the optimization domain $\tau_j \in (0,1)$ is now bounded. The choice of the Gaussian correlation is also well-justified for the application at hand, since fully developed turbulence dynamics are known to be relatively smooth.

It is supposed that simulations are run at settings $\boldsymbol{c_1}, \ldots, \boldsymbol{c_n}$ and assumed, for the time being, that model parameters are known. In invoking the conditional distribution of the multivariate normal distribution, the time-varying coefficients at a new setting $\boldsymbol{c_{new}}$ follow the distribution

$$\beta(\boldsymbol{c_{new}})|\{\beta(\boldsymbol{c_i})\}_{i=1}^{n}$$
$$\sim \mathcal{N}\left(\boldsymbol{\mu} + \left(\mathbf{T} \otimes \mathbf{r}_{\tau,new}\right)^T \left(\mathbf{T} \otimes \mathbf{R}_\tau^{-1}\right)(\beta - \mathbf{1}_n \otimes \boldsymbol{\mu}), \mathbf{T} - \left(\mathbf{T} \otimes \mathbf{r}_{\tau,new}\right)^T \left(\mathbf{T} \otimes \mathbf{R}_\tau^{-1}\right)\left(\mathbf{T} \otimes \mathbf{r}_{\tau,new}\right)\right), \tag{48}$$

where $\mathbf{r}_{\tau,new} = \left(r_\tau(\boldsymbol{c_{new}}, \boldsymbol{c_1}), \ldots, r_\tau(\boldsymbol{c_{new}}, \boldsymbol{c_n})\right)^{\boldsymbol{T}}$ and $\mathbf{R}_\tau = r_\tau\left(\boldsymbol{c_i}, \boldsymbol{c_j}\right)_{j=1}^{n}{}_{j=1}^{n}$. Using algebraic manipulations, the minimum-MSE (MMSE) predictor for $\beta(\boldsymbol{c_{new}})|\{\beta(c_i)\}_{i=1}^{n}$ and its corresponding variance is given by

$$\begin{aligned} \hat{\beta}(\boldsymbol{c}_{new}) &= \boldsymbol{\mu} + \left(\left(\mathbf{r}_{\tau,new}^T \mathbf{R}_\tau^{-1}\right) \otimes \mathbf{I}_K\right)(\beta - \mathbf{1_n} \otimes \boldsymbol{\mu}), \\ \mathbb{V}\{\beta(\boldsymbol{c}_{new})\}|\{\beta(\boldsymbol{c_i})\}_{i=1}^{n} &= (1 - \mathbf{r}_{\tau,new}^T \mathbf{R}_\tau^{-1} \mathbf{r}_{\tau,new}^T)\mathbf{T}, \end{aligned} \tag{49}$$

where $\mathbf{I}_K$ and $\mathbf{1_n}$ denote a $K \times K$ identity matrix and a 1-vectore of n elements, respectively. Substituting this into the CPOD expansion Equation (8), the predicted r-th flow variable becomes

$$\hat{Y}^{(r)}(\mathbf{x}, t; \boldsymbol{c}_{new}) = \sum_{k=1}^{K_r} \hat{\beta}^{(\boldsymbol{r})}(\boldsymbol{c}_{new}) \mathcal{M}\{\phi^{(r)}(\mathbf{x})\}, \tag{50}$$

with the associated spatiotemporal variance

$$\mathbb{V}\left\{\hat{Y}^{(r)}(\mathbf{x},t;\boldsymbol{c}_{new})\right\}|\left\{\hat{Y}^{(r)}(\mathbf{x},t;\boldsymbol{c}_i)\right\}_{i=1}^{n}$$
$$= \sum_{k=1}^{K_r}\mathbb{V}\left\{\left\{\beta_k^{(r)}(\boldsymbol{c}_{new})\right\}|\{\beta(\boldsymbol{c}_i)\}_{i=1}^{n}\right\}\left[\mathcal{M}\left\{\phi_k^{(r)}(\mathbf{x})\right\}\right]^2, \tag{51}$$

where $\phi_k^{(r)}(\mathbf{x})$ is the *k*-th CPOD mode for flow variable *r*. This holds because the CPOD modes for a fixed flow variable are orthogonal (see Section 3.3.1).

It is worth nothing that, when model parameters are known, the MMSE predictor in Equation (49) from the proposed co-kriging model (referred to as M_A) is the same as the MMSE predictor from the simpler, independent GP model with **T** diagonal (referred to as M_0). One advantage of the co-kriging model, M_A, however, is that it provides improved UQ compared to the independent model M_0, as demonstrated below. Moreover, the MMSE predictor for a derived function *g* of the flow can be quite different between M_A and M_0. This is demonstrated in the study of turbulent kinetic energy (TKE) in Section 4.5.

3.4.3.2 Common Proper Orthogonal Decomposition (CPOD) Covariance Matrix

This section briefly describes the appeal of the CPOD covariance matrix **T** from both a physical and a statistical perspective. From the underlying governing equations, it is well-known that certain dynamic behaviors are strongly *coupled* for different flow variables [62]. For example, pressure oscillation in the form of acoustic waves within an injector can induce velocity and density fluctuations. In this sense, **T** incorporates knowledge of these physical couplings within the emulator itself, with $\mathbf{T}_{ij} \gg 0$ indicating the presence of a significant coupling between modes *i* and *j*, and vice versa. The covariance selection and estimation of **T** therefore provide a data-driven way to *extract* and

rank significant flow couplings, which is interesting in itself and can be used to guide further experiments. Note that the block submatrices of **T** corresponding to the same flow variable (marked in red in Figure 11) should be diagonal, by the orthogonality of CPOD modes.

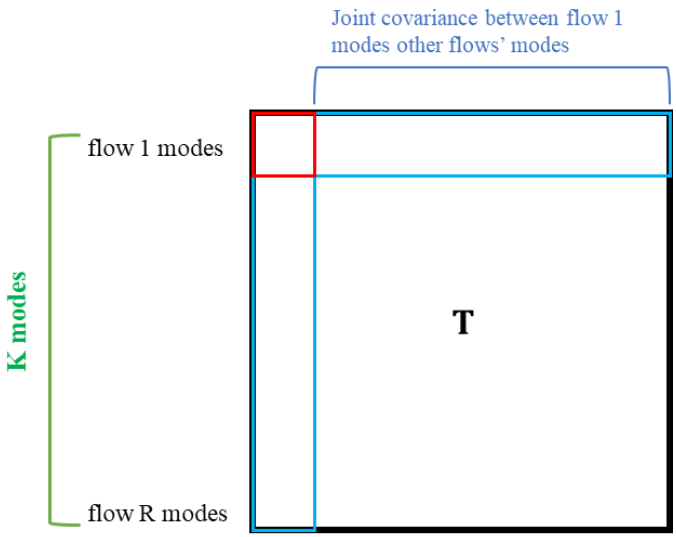

Figure 11—Illustration of the CPOD correlation matrix T (red indicates a diagonal matrix, while blue indicates non-diagonal entries)

The CPOD covariance matrix **T** also plays an important statistical role in emulation. Specifically, when significant cross-correlations exist between modes (which is known to be true from the flow couplings imposed by governing equations), the incorporation of this correlation structure within our model should provide a more accurate quantification of uncertainty. This is indeed true and is made precise by the following theorem:

Theorem 1. *Consider the two models* M_0: $\beta(\boldsymbol{c}) \in \mathbb{R}^{\boldsymbol{K}} \sim GP\{\boldsymbol{\mu}, \boldsymbol{\Sigma}^{(0)}\}$ *and* M_A: $\beta(\boldsymbol{c}) \sim GP\{\boldsymbol{\mu}, \boldsymbol{\Sigma}^{(A)}\}$, *where* $\boldsymbol{\Sigma}^{(0)}(\boldsymbol{c}_1, \boldsymbol{c}_2) = r_\tau(\boldsymbol{c}_1, \boldsymbol{c}_2)\mathbf{D}$ *and* $\boldsymbol{\Sigma}^{(A)}(\boldsymbol{c}_1, \boldsymbol{c}_2) = r_\tau(\boldsymbol{c}_1, \boldsymbol{c}_2)\mathbf{T}$ *with* $\mathbf{T} \succcurlyeq 0$ *and* $\mathbf{D} = \mathrm{diag}\{\mathbf{T}\}$. *Let* C_0 *be the* $100(1-\alpha)\%$ *highest-density confidence*

region (see Hyndman, 1996) of $\beta(\boldsymbol{c_{new}})|\{\beta(\boldsymbol{c_i})\}_{i=1}^{n}$ *under* M_0. *Suppose* $\lambda_{min}\left(\mathbf{T}^{1/2}\mathbf{D}^{-1}\mathbf{T}^{1/2}\right) > 1$. *Then,*

$$\mathbb{P}\{\beta(\boldsymbol{c}_{new}) \in C_0 | M_A, \{\beta(\boldsymbol{c}_i)_{i=1}^{n}\} < 1 - \alpha\,. \tag{52}$$

Proof*:* For brevity, let $\beta \equiv \beta(\boldsymbol{c}_{new})|\{\beta(\boldsymbol{c}_i)_{i=1}^{n}\}$, and let $\hat{\beta} \equiv \mathbb{E}[\beta(\boldsymbol{c}_{new})|\{\beta(\boldsymbol{c}_i)_{i=1}^{n}\}]$. Letting $\mathbf{Z}\sim\mathcal{N}(0, \mathbf{I}_K)$, it is shown that

$$\beta - \hat{\beta}|M_0 \sim \mathcal{N}\{0, (1 - \mathbf{r}_{\tau,new}^{T}\mathbf{R}_{\tau}^{-1}\mathbf{r}_{\tau,new}^{T})\mathbf{D}\} \stackrel{\text{def}}{=} \sqrt{1 - \mathbf{r}_{\tau,new}^{T}\mathbf{R}_{\tau}^{-1}\mathbf{r}_{\tau,new}^{T}}\,\mathbf{D}^{1/2}\mathbf{Z}\,, \text{ and}$$

$$\beta - \hat{\beta}|M_A \sim \mathcal{N}\{0, (1 - \mathbf{r}_{\tau,new}^{T}\mathbf{R}_{\tau}^{-1}\mathbf{r}_{\tau,new}^{T})\mathbf{T}\} \stackrel{\text{def}}{=} \sqrt{1 - \mathbf{r}_{\tau,new}^{T}\mathbf{R}_{\tau}^{-1}\mathbf{r}_{\tau,new}^{T}}\,\mathbf{T}^{1/2}\mathbf{Z}\,.$$

Under the independent model M_0, the $100(1-\alpha)\%$ highest-density confidence region becomes

$$C_0 = \left\{\boldsymbol{\xi} : \left(1 - \mathbf{r}_{\tau,new}^{T}\mathbf{R}_{\tau}^{-1}\mathbf{r}_{\tau,new}^{T}\right)^{-1}\left(\boldsymbol{\xi} - \hat{\beta}\right)^{\mathbf{T}}\mathbf{D}^{-1}\left(\boldsymbol{\xi} - \hat{\beta}\right) \le \chi_K^2(1-\alpha)\right\},$$

where $\chi_K^2(1-\alpha)$ is the $(1-\alpha)$-quantile of a χ^2-distribution with K degrees of freedom. Now, let λ_{min} denote the minimum eigenvalue of $\mathbf{T}^{1/2}\mathbf{D}^{-1}\mathbf{T}^{1/2}$. It follows that

$$\begin{aligned}\mathbb{P}(\beta \in C_0|M_A) &= \mathbb{P}\left\{\left(\beta - \hat{\beta}\right)^{T}\mathbf{D}^{-1}\left(\beta - \hat{\beta}\right) \le \left(1 - \mathbf{r}_{\tau,new}^{T}\mathbf{R}_{\tau}^{-1}\mathbf{r}_{\tau,new}^{T}\right)\chi_K^2(1-\alpha)|M_A\right\} \\ &= \mathbb{P}\left\{\mathbf{Z}^{T}\left(\mathbf{T}^{1/2}\mathbf{D}^{-1}\mathbf{T}^{1/2}\right)\mathbf{Z} \le \chi_K^2(1-\alpha)\right\} \\ &\le \mathbb{P}\{\mathbf{Z}^{T}\mathbf{Z} \le \lambda_{min}^{-1}\chi_K^2(1-\alpha)\},\end{aligned} \tag{53}$$

since it is almost sure that $\mathbf{Z}^T\left(\mathbf{T}^{1/2}\mathbf{D}^{-1}\mathbf{T}^{1/2}\right)\mathbf{Z} \geq \lambda_{min}\mathbf{Z}^T\mathbf{Z}$. The asserted result follows because $\mathbb{P}\{\mathbf{Z}^T\mathbf{Z} \leq \lambda_{min}^{-1}\chi_K^2(1-\alpha)\}$ is strictly less than $1-\alpha$ when $\lambda_{min} > 1$.

□

In other words, this theorem quantifies the effect on coverage probability when the true co-kriging model M_A, which accounts for cross-correlations between modes, is mis-specified as M_0, the independent model ignoring such cross-correlations. Note that an increase in the number of significant non-zero cross-correlations in $\mathbf{T}$ causes $\mathbf{T}^{1/2}\mathbf{D}^{-1}\mathbf{T}^{1/2}$ to deviate further from unity, which in turn may increase λ_{min}. Given enough such correlations, **Theorem 1** shows that the coverage probability from the mis-specified model M_0 is less than the desired $100(1-\alpha)\%$ rate. In the CPOD case, this suggests that when sufficient significant flow couplings exist, the co-kriging model M_A provides more accurate UQ for the *joint* prediction of flow variables, when compared to the mis-specified independent model M_0. This improvement also holds for functions of flow variables (as is demonstrated in Section 4.5), although a formal argument is not presented here.

It is important to identify an important trade-off for co-kriging models in general and outline why the proposed model is appropriate for the current application in view of this trade-off. The spatial statistics literature [86, 87] demonstrates that when the matrix $\mathbf{T}$ exhibits strong correlations and can be well-estimated, it enjoys improved predictive performance through a co-kriging model (this is formally shown for the current model in **Theorem 1**). However, when such correlations are absent or cannot be well-estimated, a co-kriging model can yield poorer performance than an independent model. This study posits that the former is true for the current application. First, the differential equations

governing the simulation procedure explicitly impose strong dependencies between flow variables, providing *a priori* the existence of strong correlations in $\mathbf{T}$. Second, it is demonstrated in Section 4.4 that the dominant correlations selected in **T** are physically interpretable in terms of fluid mechanic principles and conservation laws, which provides strong evidence for the correct estimation of **T**.

An issue with fitting M_A is that many more parameters must be estimated. Specifically, since the CPOD covariance matrix **T** is $K \times K$ dimensional, there is insufficient data for estimating all entries in **T** using the extracted coefficients from the CPOD expansion. A solution is to impose the sparsity constraint $\left\|\mathbf{T^{-1}}\right\|_1 \leq \gamma$, where $\|A\|_1 = \sum_{k=1}^{K}\sum_{l=1}^{K}|A_{kl}|$ is the element-wise L_1 norm. For a small choice of γ, this forces nearly all entries in $\mathbf{T^{-1}}$ to be zero, thus permitting consistent estimation of the few significant correlations. Sparsity can also be justified from an engineering perspective, because the number of significant couplings is known to be small from flow physics. In addition, γ can be adjusted to extract a pre-specified number of flow couplings, which is appealing from an engineering point of view. This justification for sparsifying $\mathbf{T^{-1}}$ instead of **T** is largely computational, because algorithmically, the former problem can be handled much more efficiently than the latter using the graphical lasso (i.e., least absolute shrinkage and selection operator, also Lasso or LASSO) [88, 89]. Such efficiency is crucial here, since GP parameters must be jointly estimated as well.

Although the proposed model is similar to the one developed by Qian et al. (2008)[90] for emulating qualitative factors, there are two key distinctions. First, our model allows for different process variances for each coefficient, whereas their approach restricts

all coefficients to have equal variances. Second, our model incorporates sparsity on the CPOD covariance matrix, an assumption necessary from a statistical point of view and appealing from a physics extraction perspective. Lastly, the algorithm proposed below can estimate **T** more efficiently than their semidefinite programming approach [90].

3.4.3.3 Parameter Estimation

To estimate the model parameters $\boldsymbol{\mu}$, **T**, and $\boldsymbol{\tau}$, MLE is used in favor of a Bayesian implementation. This choice is made for computational efficiency; for the proposed emulator to be used as a quick investigative tool for surveying the design space, it should generate flow predictions much quicker than a direct simulation, which requires several days to complete parallelized computation. From Equation (46) and Equation (47), the maximum-likelihood formulation can be written as $\text{argmin}_{\boldsymbol{\mu},\mathbf{T},\boldsymbol{\tau}}\ l_\lambda(\boldsymbol{\mu},,\boldsymbol{\tau})$, where $l_\lambda(\boldsymbol{\mu},\mathbf{T},\boldsymbol{\tau})$ is the penalized negative log-likelihood:

$$l_\lambda(\boldsymbol{\mu},\mathbf{T},\boldsymbol{\tau}) = n\log\det\mathbf{T} + K\log\det\mathbf{R}_\tau + (\mathbf{B}-\mathbf{1}_n\otimes\boldsymbol{\mu})^T[\mathbf{R}_\tau\otimes\mathbf{T}^{-1}](\mathbf{B}-\mathbf{1}_n\otimes\boldsymbol{\mu}) + \lambda\|\mathbf{T}^{-1}\|_1 \quad \textbf{(54)}$$

Note that, because the formulation is convex in $\mathbf{T}^{-1}$, the sparsity constraint $\|\mathbf{T}^{-1}\|_1 \le \gamma$ has been incorporated into the likelihood through the penalty $\lambda\|\mathbf{T}^{-1}\|_1$, using strong duality. As with γ, a large λ results in a smaller number of selected correlations, and vice versa. The tuning method for λ depends on the desired end-goal. For example, if predictive accuracy is the primary goal, λ should be tuned using cross-validation techniques [79]. However, if correlation extraction is desired or prior information is available on flow

couplings, then λ should be set so that a fixed (preset) number of correlations is extracted. This is discussed further in Section 4.5.

A fixed penalty $\lambda > 0$ is assumed. To compute the MLEs in Equation (54), the following *blockwise coordinate descent* algorithm is proposed. First, initial values are assigned for $\boldsymbol{\mu}$, $\mathbf{T}$, and $\boldsymbol{\tau}$. Next, the following two updates are iterated until parameters converge: (a) for fixed GP parameters $\boldsymbol{\mu}$ and $\boldsymbol{\tau}$, optimized for $\mathbf{T}$ in Equation (54) and (b) for fixed covariance matrix $\mathbf{T}$, optimized for $\boldsymbol{\mu}$ and $\boldsymbol{\tau}$ in Equation (54). With the use of the graphical LASSO algorithm discussed by Friedman et al. (2008) [88], the first update can be computed efficiently. The second update can be computed using nonlinear optimization techniques on $\boldsymbol{\tau}$ by means of a closed-form expression for $\boldsymbol{\mu}$. In the current implementation, this is performed using the L-BFGS algorithm [69], which offers a super-linear convergence rate without the cumbersome evaluation and manipulation of the Hessian matrix [91]. The following theorem guarantees that the proposed algorithm converges to a stationary point of Equation (54) (see [1] for proof):

Theorem 2. *The Blockwise Coordinate Descent scheme in Algorithm 1converges to some solution* $(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{T}}, \hat{\boldsymbol{\tau}})$*, which is stationary for the penalized log-likelihood* $\boldsymbol{l_\lambda(\mu, T, \tau)}$*. [1]*

The proof of Theorem 2 can be found in Mak et al. [1]. It is worth noting that the proposed algorithm does not provide global optimization. This is not surprising, since the log-likelihood l_λ is non-convex in $\boldsymbol{\tau}$. To this end, multiple threads of Algorithm 1 are run in parallel, each with a different initial point $\boldsymbol{\tau_0}$ from a large space-filling design on $[10^{-3}, 1 - 10^{-3}]^p$, then the converged parameter setting is chosen that yields the largest

likelihood value from Equation (54). In our experience, this heuristic performs quite well in practice.

Algorithm 4—Blockwise coordinate descent algorithm for maximum-likelihood estimation (MLE) [1]

1: **for** each time step $\boldsymbol{t} = \mathbf{1}, \ldots, \boldsymbol{T}$ **do parallel**

2: • Set initial values $\boldsymbol{\mu} \leftarrow \mathbf{0}_K$, $\mathbf{T} \leftarrow \mathbf{I}_K$ and $\boldsymbol{\tau} \leftarrow \mathbf{I}_p$, and set $\mathbf{B} \leftarrow (\beta(\mathbf{c}_1), \ldots, \beta(\mathbf{c}_n))^T$

3: **repeat**

4: Optimizing **T**:

5: • Set $\mathbf{W} \leftarrow \frac{1}{n}(\mathbf{B} - \mathbf{1}_n \otimes \boldsymbol{\mu}^T)^T \mathbf{R}_\tau^{-1}(\mathbf{B} - \mathbf{1}_n \otimes \boldsymbol{\mu}^T) + \lambda \cdot \mathbf{I}_K$

6: **repeat**

7: for $j = 1, \ldots, K$ do

8: • Solve $\widetilde{\boldsymbol{\delta}} = \text{argmin}_{\boldsymbol{\delta}} \left\{ \frac{1}{2} \left\| \mathbf{W}_{-j,-j}^{1/2} \boldsymbol{\delta} \right\|_2^2 + \lambda \|\boldsymbol{\delta}\|_1 \right\}$ using LASSO

9: • Update $\mathbf{W}_{-j,j} \leftarrow \mathbf{W}_{-j,-j} \widetilde{\boldsymbol{\delta}}$ and $\mathbf{W}_{j,-j}^T \leftarrow \mathbf{W}_{-j,-j}^{1/2} \widetilde{\boldsymbol{\delta}}$

10: **until W** converges

11: • Update $\mathbf{T} \leftarrow \mathbf{W}^{-1}$

12: Optimizing $\boldsymbol{\mu}$ and $\boldsymbol{\tau}$:

13: • Update $\boldsymbol{\tau} \leftarrow \text{argmin}_{\boldsymbol{\mu},\mathbf{T},\boldsymbol{\tau}} l_\lambda(\boldsymbol{\mu}, \mathbf{T}, \boldsymbol{\tau})$ with L-BFGS, with $\boldsymbol{\mu}_{\boldsymbol{\tau}} = (\mathbf{1}_n^T \mathbf{R}_\tau^{-1} \mathbf{1}_n)^T (\mathbf{1}_n^T \mathbf{R}_\tau^{-1} \mathbf{B})$

14: • Update $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu}_{\boldsymbol{\tau}}$

15: **until** $\boldsymbol{\mu}$**,** **T,** and $\boldsymbol{\tau}$ converges

16: **end parallel for**

17: • **return** $\boldsymbol{\mu}(t)$, $\mathbf{T}(t)$, and $\boldsymbol{\tau}(t)$,

---

3.4.3.4 Kernel-Smoothed Proper Orthogonal Decomposition (KSPOD) Model Specifications

The KSPOD model, after the POD data reduction, functions identically to CPOD in that the extracted time-varying coefficients $\{\beta_k(t, \boldsymbol{c}_i)\}_{i,k}$ are used as data for fitting the proposed emulator. The model is as follows. The KSPOD expansion in Equation (18) is truncated at k terms for flow $r = 1, \dots, \mathcal{R}$ (with $\mathcal{R} = 30$ in the present case). It is assumed that $\beta^r(t; \boldsymbol{c}) = \left(\beta_k^r(t; \boldsymbol{c}), \dots, \beta_k^r(t; \boldsymbol{c})\right)^{\mathrm{T}}$ is the vector of $\mathcal{K}$ time-varying coefficients for flow variable r at design setting $\boldsymbol{c}$, with $\beta(t; \boldsymbol{c}) = \left(\beta_k^{(1)}(t; \boldsymbol{c}), \dots, \beta_k^{(\mathcal{R})}(t; \boldsymbol{c})\right)^{\mathrm{T}}$ being the coefficient vector for all flows at $\boldsymbol{c}$. The following *time-independent GP model* on $\beta(t; \boldsymbol{c})$ is assumed:

$$\beta(t; \boldsymbol{c}) \sim GP\{\boldsymbol{\mu}(t), \boldsymbol{\Sigma}(\cdot, \cdot; t)\}, \beta(t; \boldsymbol{c}) \perp \beta(t'; \boldsymbol{c}) \text{ for } t \neq t'. \tag{55}$$

Here, $\mathcal{K} = k \times \mathcal{R}$ is the number of extracted modes over all R flow variables, $\boldsymbol{\mu} \in \mathbb{R}^{\boldsymbol{\mathcal{K}}}$ is the process mean vector, and $\Sigma(\cdot,\cdot) : \mathbb{R}^{\boldsymbol{k}} \times \mathbb{R}^{\boldsymbol{k}} \to \mathbb{R}^{\boldsymbol{\mathcal{K}} \times \boldsymbol{\mathcal{K}}}$ is its corresponding covariance matrix function, as defined below. Since the GPs are now time-independent, the specification for *fixed* time t is presented, and $\beta(t; \boldsymbol{c})$, $\boldsymbol{\mu}(t)$ and $\boldsymbol{\Sigma}(\cdot, \cdot; t)$ is referred to as $\beta(\boldsymbol{c})$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}(\cdot, \cdot)$ for brevity. For computational efficiency, the following separable form is assumed for $\boldsymbol{\Sigma}(\cdot, \cdot)$:

$$\boldsymbol{\Sigma}(\boldsymbol{c_1}, \boldsymbol{c_2}) = r_\tau(\boldsymbol{c_1}, \boldsymbol{c_2})\mathbf{T}^*, \; r_\tau(\boldsymbol{c_1}, \boldsymbol{c_2}) = \prod_{j=1}^{p} \tau_j^4 \left(c_{1j} - c_{2j}\right)^2, \tag{56}$$

$$\boldsymbol{c_1}, \boldsymbol{c_2} \in \mathbb{R}^{\boldsymbol{p}}, \tau_j \in (0,1),$$

where $\mathbf{T}^* \in \mathbb{R}^{\mathcal{K}\times\mathcal{K}}$ is a symmetric, positive definite matrix that diagonalizes covariance matrices from all training cases, and $\tau_r(\cdot\,,\cdot\,)$ is the correlation function over the design space, parameterized by $\boldsymbol{\tau} = \left(\tau_1, \ldots, \tau_p\right)^T \in (0,1)^p$. It is noted that $\mathbf{T}^*$ in Figure 12 differs from the CPOD covariance shown in Figure 11. This can be viewed as a large co-kriging model [85] over the design space, with the multivariate observations being the extracted CPOD coefficients for all flow variables. Note that r_τ is a reparameterization of the squared-exponential (or Gaussian) correlation function $\exp\{-\sum_{j=1}^{p}\theta_j\left(c_{1j}-c_{2j}\right)^2\}$, with $\theta_j = -4\log\tau_j$. In our experience, such a reparameterization allows for a more numerically stable optimization of MLEs, because the optimization domain $\tau_j \in (0,1)$ is now bounded. The choice of the Gaussian correlation is also well-suited for the application at hand, since fully developed turbulence dynamics are known to be relatively smooth.

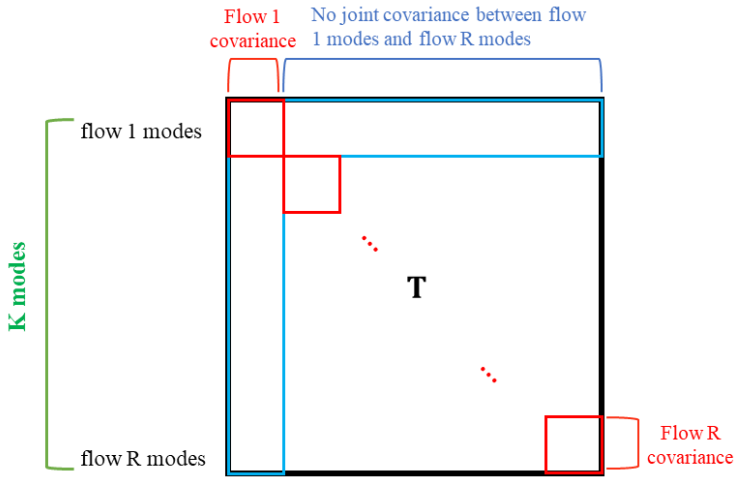

Figure 12—Illustration of the KSPOD correlation matrix $\mathbf{T}^*$ (red indicates a diagonal matrix, while blue indicates non-diagonal entries)

It is supposed that simulations are run at settings $\boldsymbol{c_1}, \ldots, \boldsymbol{c_n}$, and assumed for the time being that model parameters are known. Invoking the conditional distribution of the

multivariate normal distribution, the time-varying coefficients at a new setting $\boldsymbol{c_{new}}$ follows the distribution

$$\beta(\boldsymbol{c_{new}})|\{\beta(\boldsymbol{c_i})\}_{i=1}^{n}$$
$$\sim \mathcal{N}\left(\boldsymbol{\mu} + \left(\mathbf{T}^* \otimes \mathbf{r}_{\tau,new}\right)^T \left(\mathbf{T}^* \otimes \mathbf{R}_\tau^{-1}\right)(\beta - \mathbf{1}_n \otimes \boldsymbol{\mu}), \mathbf{T}^* - \left(\mathbf{T}^* \otimes \mathbf{r}_{\tau,new}\right)^T \left(\mathbf{T}^* \otimes \mathbf{R}_\tau^{-1}\right)\left(\mathbf{T} \otimes \mathbf{r}_{\tau,new}\right)\right), \tag{57}$$

where $\mathbf{r}_{\tau,new} = \left(r_\tau(\boldsymbol{c_{new}}, \boldsymbol{c_1}), \dots, r_\tau(\boldsymbol{c_{new}}, \boldsymbol{c_n})\right)^T$ and $\mathbf{R}_\tau = r_\tau\left(\boldsymbol{c_i}, \boldsymbol{c_j}\right)_{j=1\ j=1}^{n\quad n}$. Using algebraic manipulations, the MMSE predictor for $\beta(\boldsymbol{c_{new}})|\{\beta(c_i)\}_{i=1}^{n}$ and its corresponding variance is given by

$$\begin{aligned} \hat{\beta}(\boldsymbol{c}_{new}) &= \boldsymbol{\mu} + \left(\left(\mathbf{r}_{\tau,new}^T \mathbf{R}_\tau^{-1}\right) \otimes \mathbf{I}_K\right)(\beta - \mathbf{1_n} \otimes \boldsymbol{\mu}), \\ \mathbb{V}\{\beta(\boldsymbol{c}_{new})\}|\{\beta(\boldsymbol{c_i})\}_{i=1}^{n} &= (1 - \mathbf{r}_{\tau,new}^T \mathbf{R}_\tau^{-1} \mathbf{r}_{\tau,new}^T)\mathbf{T}^*, \end{aligned} \tag{58}$$

where $\mathbf{I}_K$ and $\mathbf{1_n}$ denote a $K \times K$ identity matrix and a 1-vectore of n elements, respectively. The same method can be applied to the weighting number in Equation (19). Substituting this into the KSPOD expansion Equation (18), the predicted r-th flow variable becomes

$$\begin{aligned} \hat{Y}^{(r)}(\boldsymbol{x}, t; \boldsymbol{c}_{new}) &= \sum_{k=1}^{K_r} \hat{\beta}_k^{(r)}(\boldsymbol{c}_{new}) \hat{\phi}_k^{(r)}(\boldsymbol{c}_{new}, \boldsymbol{x}) \\ &= \sum_{k=1}^{K_r} \hat{\beta}^{(r)}(\boldsymbol{c}_{new}) \frac{\sum_{i=1}^{n} \widehat{w}_i(\boldsymbol{c}_{new}) \phi_k^{(r)}(\boldsymbol{c}_i, \boldsymbol{x}_j)}{\sum_{i=1}^{n} \widehat{w}_i(\boldsymbol{c}_{new})} \end{aligned} \tag{59}$$

$$= \sum_{k=1}^{K_r} \hat{\beta}^{(r)}(\boldsymbol{c}_{new}) \left[\mathcal{W}(\boldsymbol{c}_{new}) \phi_k^{(r)}(\boldsymbol{c}_i, \boldsymbol{x}_j) \right],$$

with the associated spatiotemporal variance

$$\begin{aligned}
&\mathbb{V}\{\hat{Y}^{(r)}(\boldsymbol{x}, t; \boldsymbol{c}_{new})\} | \{\hat{Y}^{(r)}(\boldsymbol{x}, t; \boldsymbol{c}_i)\}_{i=1}^n \\
&= \sum_{k=1}^{K_r} \mathbb{V}\left\{ \left\{ \beta_k^{(r)}(\boldsymbol{c}_{new}) \right\} | \{\beta(\boldsymbol{c}_i)\}_{i=1}^n \right\} \left[\hat{\phi}_k{}^{(r)}(\boldsymbol{c}_{new}, \boldsymbol{x}) \right]^2 \\
&= \sum_{k=1}^{K_r} \mathbb{V}\left\{ \left\{ \beta_k^{(r)}(\boldsymbol{c}_{new}) \right\} | \{\beta(\boldsymbol{c}_i)\}_{i=1}^n \right\} \mathbb{V}\{\{\hat{w}_i(\boldsymbol{c}_{new}))\} | \{\hat{w}_i(\boldsymbol{c}_{new})\}_{i=1}^n \} \left[\phi_k^{(r)}(\boldsymbol{c}_i, \boldsymbol{x}_j) \right]^2,
\end{aligned} \tag{60}$$

where $\phi_k^{(r)}$ is the *k*-th KSPOD mode for flow variable *r*, and $\hat{w}_i$ is the weighing number for KSPOD modes. This holds because the KSPOD modes for a fixed flow variable are orthogonal. Once the variance of predictions is calculated, the co-kriging model $M_{A,}$ proposed for UQ analysis, can be applied as well.

3.4.3.5 Common Kernel-Smoothed Proper Orthogonal Decomposition (CKSPOD) Model Specifications

The CKSPOD model combines CPOD and KSPOD. Hence, the definition of variance for this emulation model resembles the other two emulators. It is noted that in CKSPOD, a transfer matrix $\mathcal{T}$ in Equation (34) is created from the CKSPOD covariance $\mathbb{C}$ in Equation (32) and Figure 13. Hence, even the CKSPOD expansion in Equation (46) is truncated at k terms for flow $r = 1, \dots, \mathcal{R}$ (with $\mathcal{R} = 30$ in the present case). It is assumed that $\beta^r(t; \boldsymbol{c}) = \left(\beta_k^r(t; \boldsymbol{c}), \dots, \beta_k^r(t; \boldsymbol{c})\right)^{\mathrm{T}}$ is the vector of k time-varying coefficients for

flow variable r at design setting $\boldsymbol{c}$ and $\beta(t;\boldsymbol{c}) = \left(\beta_k^{(1)}(t;\boldsymbol{c}), \ldots, \beta_k^{(\mathcal{R})}(t;\boldsymbol{c})\right)^{\mathrm{T}}$ is the coefficient vector for all flows at $\boldsymbol{c}$. Using algebraic manipulations, the MMSE predictor for $\beta(\boldsymbol{c_{new}})|\{\beta(c_i)\}_{i=1}^n$ and its corresponding variance is given by

$$\begin{aligned}\hat{\beta}(\boldsymbol{c}_{new}) &= \boldsymbol{\mu} + \left(\left(\mathbf{r}_{\tau,new}^T \mathbf{R}_\tau^{-\mathbf{1}}\right) \otimes \mathbf{I}_K\right)(\beta - \mathbf{1_n} \otimes \boldsymbol{\mu}),\\ \mathbb{V}\left\{\tilde{\beta}(\boldsymbol{c}_{new})\right\}|\left\{\tilde{\beta}(\boldsymbol{c_i})\right\}_{i=1}^n &= (1 - \mathbf{r}_{\tau,new}^T \mathbf{R}_\tau^{-\mathbf{1}} \mathbf{r}_{\tau,new}^T)\mathbb{C},\end{aligned} \tag{61}$$

where $\mathbf{I}_K$ and $\mathbf{1_n}$ denote a $k \times k$ identity matrix and a 1-vectore of n elements, respectively. Substituting this into the CKSPOD expansion Equation (46) and the weighting number for modes, the predicted r-th flow variable becomes

$$\hat{Y}^{(r)}(\boldsymbol{x}, t; \boldsymbol{c}_{new}) = \sum_{k=1}^{K_r} \tilde{\beta}_k^{(r)}(\boldsymbol{c}_{new}, t)\left[\frac{\sum_{i=1}^n \widehat{w}_i(\boldsymbol{c}_{new})\tilde{\phi}_k^{(r)}(\boldsymbol{c}_i, x_j)}{\sum_{i=1}^n \widehat{w}_i(\boldsymbol{c}_{new})}\right] \tag{62}$$

with the associated spatiotemporal variance

$$\begin{aligned}&\mathbb{V}\left\{\hat{Y}^{(r)}(\boldsymbol{x}, t; \boldsymbol{c}_{new})\right\}|\left\{\hat{Y}^{(r)}(\boldsymbol{x}, t; \boldsymbol{c}_i)\right\}_{i=1}^n\\ &= \sum_{k=1}^{K_r} \mathbb{V}\left\{\left\{\tilde{\beta}_k^{(r)}(\boldsymbol{c}_{new})\right\}|\left\{\tilde{\beta}(\boldsymbol{c}_i)\right\}_{i=1}^n\right\}\left[\hat{\phi}_k{}^{(r)}(\boldsymbol{c}_{new}, \boldsymbol{x})\right]^2\\ &= \sum_{k=1}^{K_r} \mathbb{V}\left\{\left\{\tilde{\beta}_k^{(r)}(\boldsymbol{c}_{new})\right\}|\left\{\tilde{\beta}(\boldsymbol{c}_i)\right\}_{i=1}^n\right\}\mathbb{V}\left\{\{\widehat{w}_i(\boldsymbol{c}_{new}))\}|\{\widehat{w}_i(\boldsymbol{c}_{new})\}_{i=1}^n\right\}\left[\tilde{\phi}_k^{(r)}(\boldsymbol{c}_i, x\ \ )\right]^2,\end{aligned} \tag{63}$$

where $\tilde{\phi}_k^{(r)}(\boldsymbol{c}_i, x_j)$ is the *k*-th CKSPOD mode in for flow variable *r* and $\hat{w}_i$ is the weighing number for CKSPOD modes.

In closing, this dissertation employs a great amount of simulation data sets as the data vault as the foundation to build a time-efficient surrogate model to predict spatiotemporal evolving flow. The emulation model is not easy to build. This chapter shows the evolution of development of the three methodologies (CPOD, KSPOD, and CKSPOD). The simulation results will be addressed and the emulation results will be demonstrated and analyzed carefully in the following chapter.

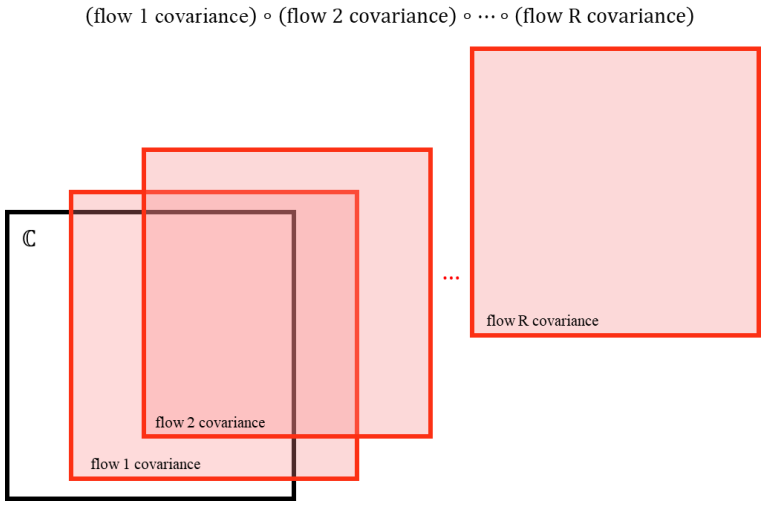

Figure 13—Illustration of the CKSPOD covariance matrix ℂ (all flow information is compiled by the Hadamard product)

CHAPTER 4. RESEARCH RESULTS AND DISCUSSION

This chapter demonstrates and discusses the results of nonreacting flow in a simplex swirl injector from three emulator models. Section 4.1 shows the high-fidelity simulation results which are used as data vault to train the emulators in this dissertation. Section 4.2 elaborates on the research discoveries regarding CPOD, including details of statistical and physics analysis. Section 3.2 states the results of KSPOD and the accuracy of analysis. Section 4.4 elaborates upon the spatiotemporal evolving flow prediction results based on the latest emulator model, CKSPOD. This section also interprets the prediction differences among the three emulators. Section 4.5 proposes a UQ analysis model to estimate the precision of all proposed emulation models.

4.1 High-Fidelity Simulation Results

4.1.1 Nonreacting Flows for MaxPro Cases

Thirty high-fidelity simulations for CPOD at design points defined by MaxPro were conducted. To isolate the effect of injector parameters, the mass flow rate for all runs is fixed at 0.15 kg/s. The first two design points designated by MaxPro are chosen as the baseline geometries, A and B in Table 8. The benchmark points used for assessing the accuracy of the emulator model are obtained by offsetting the design parameters of these two points.

Figures 14 and 15 show the instantaneous distributions of temperature and density for two neighboring design points, C and D in Table 8, selected to indicate different flow features in the design space. The key flow structures include the swirling liquid film along

the wall due to centrifugal force, liquid accumulation near the injector headend and associated flow recirculation, and a conical liquid sheet spreading outward at the injector exit propelled by azimuthal momentum and a hollow gas core in the center region [46].

Table 8—Injector geometrics at design points colored blue in Figure 7

Design	L (mm)	R_n (mm)	θ (°)	δ (mm)	ΔL (mm)
A (swirl)	20.0	3.22	52.9	0.52	3.42
B (jet-like)	41.9	3.05	65.5	1.57	1.00
C (swirl)	43.1	5.00	70.0	0.50	2.79
D (jet-like)	37.7	2.82	45.8	1.17	3.80

Various flow physics are observed. The film thickness for Design C is much thinner than for Design D, with a larger spreading angle at the injector exit (34.6° compared to 29.2° for Design D). Among the 30 design points, some behave like swirling flows, as in Design C, while others behave like jet flows, as in Design D. For convenience, the critical value of the spreading angle that separates swirling from jet-like flows is chosen to be 30°; this angle is considered an empirical indicator of whether the liquid stream has significant radial penetration in the downstream region. When this angle is not achieved, the liquid does not have enough radial momentum to spread outwards. The 30 simulation runs are thus divided into two subgroups: swirling flows (spreading angle above 30°) and jet-like flows (spreading angle below 30°). In the following section, a machine-learning technique, decision tree, is introduced to identify the jet-swirl dichotomy. This directly influences the feature extraction and kriging processes described in the following sections, as it changes how the design space is partitioned between the identified flow behaviors. Implicitly, the

extracted coherent structures will change slightly, depending on the established criteria separating swirling and jet-like flows.

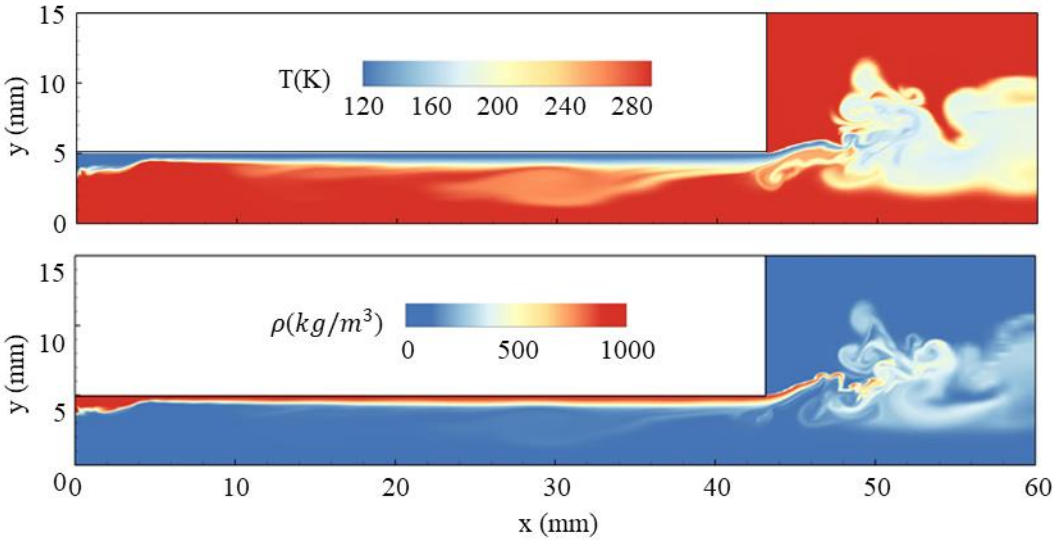

Figure 14—Instantaneous distributions of temperature and density for Design C

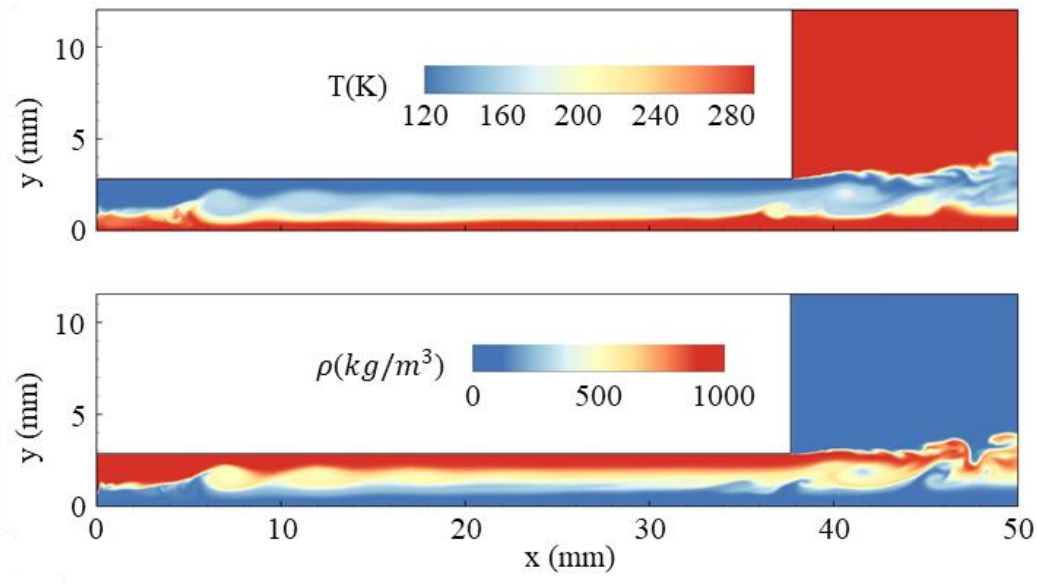

Figure 15—Instantaneous distributions of temperature and density for Design D

4.1.2 Nonreacting Flows for SLHD Cases

The LES-based high-fidelity numerical framework described previously was implemented for the 30 cases selected by SLHD for the three design parameters, which are decided by the sensitivity analysis using a first-order Monte Carlo estimation of Sobol indices [2]. The sensitivity analysis was performed based on the desired liquid-film thickness and spreading angle. The three chosen parameters are significantly influenced by the inlet velocity, u_{in}, which ranges from 5.71 to 40.43 m/s, as listed in Table 9. The 30 training cases are roughly classified into four groups in terms of u_{in} (m/s), as follows: Cluster A, with $u_{in} < 10$; Cluster B, with $10 \leq u_{in} < 18$; Cluster C, with $18 \leq u_{in} < 25$; and Cluster D, with $u_{in} > 25$.

Overall, there are 15 cases in Cluster A, 8 in Cluster B, 5 in Cluster C, and 2 in Cluster D. Figures 16-19 show two sample snapshots of the density field for each cluster. Variations of film thickness and spreading angle are observed. Cluster A has the slowest inlet velocity and generates thicker film and smaller spreading angle. The film thickness decreases and the spreading angle increases between Clusters B and C; the smallest film thickness and largest spreading angle appear in Cluster D, where the highest inlet velocity exists. As previously mentioned, the geometric constant, K, is an indicator of swirl strength, with a higher value implying stronger azimuthal momentum. When K becomes small, the swirling flow does not contain enough azimuthal momentum to produce a large spreading angle of the liquid film. This suggests that K plays a pivotal role, with a slight change causing significant differences in flow dynamics. Therefore, the training process must preserve important flow physics during data reduction.

Table 9—Design matrix and associated inlet velocity information

Case	δ (mm)	θ (deg)	ΔL (mm)	u_{in} (m/s)	u_r (m/s)	u_θ (m/s)	K	*Cluster*
1	0.28	57.92	1.59	40.43	21.47	34.26	7.44	D
2	0.63	40.81	1.93	12.35	9.35	8.07	1.64	B
3	0.82	52.39	0.96	11.79	7.20	9.34	1.98	B
4	1.10	32.76	2.57	6.42	5.40	3.47	0.69	A
5	1.12	51.88	3.21	8.58	5.30	6.75	1.43	A
6	1.52	46.85	2.23	5.71	3.90	4.16	0.86	A
7	0.38	37.29	1.64	19.53	15.54	11.83	2.37	C
8	0.51	52.89	2.15	19.35	11.67	15.43	3.27	C
9	0.78	43.33	3.12	10.43	7.58	7.15	1.46	B
10	1.03	33.76	0.87	6.89	5.73	3.83	0.76	A
11	1.26	49.37	1.72	7.19	4.68	5.46	1.14	A
12	1.39	60.44	2.61	8.63	4.26	7.51	1.65	A
13	0.47	54.40	2.74	21.87	12.73	17.78	3.80	C
14	0.68	38.80	2.53	11.25	8.77	7.05	1.42	B
15	0.74	48.36	1.89	12.06	8.02	9.02	1.88	B
16	0.93	33.26	1.47	7.63	6.38	4.18	0.83	A
17	1.22	42.82	0.91	6.60	4.84	4.49	0.92	A
18	1.35	57.42	3.17	8.15	4.39	6.87	1.49	A
19	0.32	58.43	2.27	35.58	18.63	30.31	6.60	D
20	0.59	34.77	1.13	12.19	10.01	6.95	1.38	B
21	0.84	49.87	2.83	10.89	7.02	8.32	1.74	B
22	0.99	44.33	1.76	8.35	5.97	5.84	1.20	A
23	1.20	37.79	3.08	6.24	4.93	3.82	0.77	A
24	1.45	55.41	1.55	7.17	4.07	5.90	1.27	A
25	0.40	36.28	2.32	18.27	14.73	10.81	2.16	C
26	0.49	51.38	1.42	19.51	12.18	15.24	3.21	C
27	0.72	53.39	3.29	13.84	8.25	11.11	2.36	B
28	0.95	40.31	1.17	8.18	6.24	5.29	1.07	A
29	1.24	59.43	1.98	9.36	4.76	8.06	1.76	A
30	1.37	43.83	2.78	5.99	4.32	4.15	0.85	A

Note: Cases 1-6 are on Slice 1, Cases 7-12 on Slice 2, Cases 13-18 on Slice 3, Cases 19-24 on Slice 4, and Cases 25-30 on Slice 5, corresponding to the symbols on Figure 2.

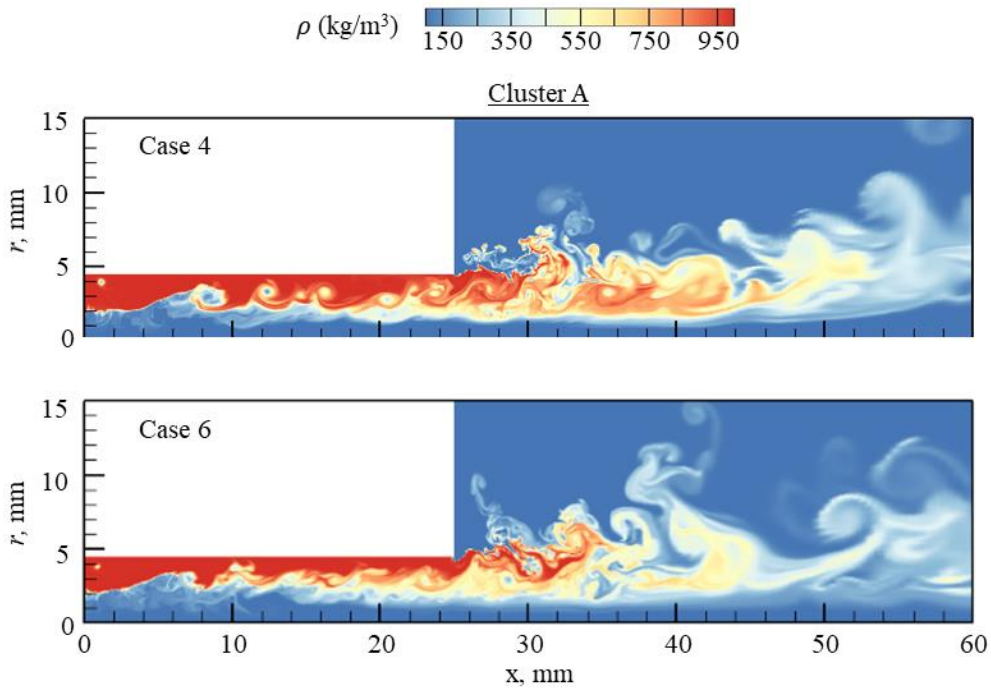

Figure 16—Selected snapshots of density field in Cluster A at *t* = 1.21 ms, obtained from LES-based high-fidelity simulations

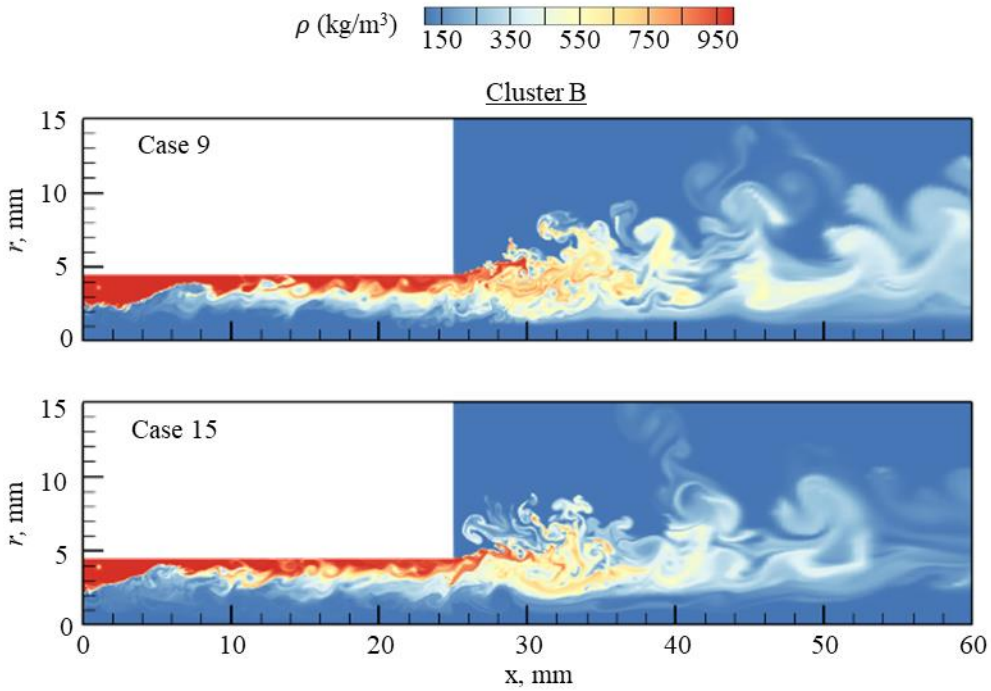

Figure 17—Selected snapshots of density field in Cluster B at *t* = 1.21 ms, obtained from LES-based high-fidelity simulations

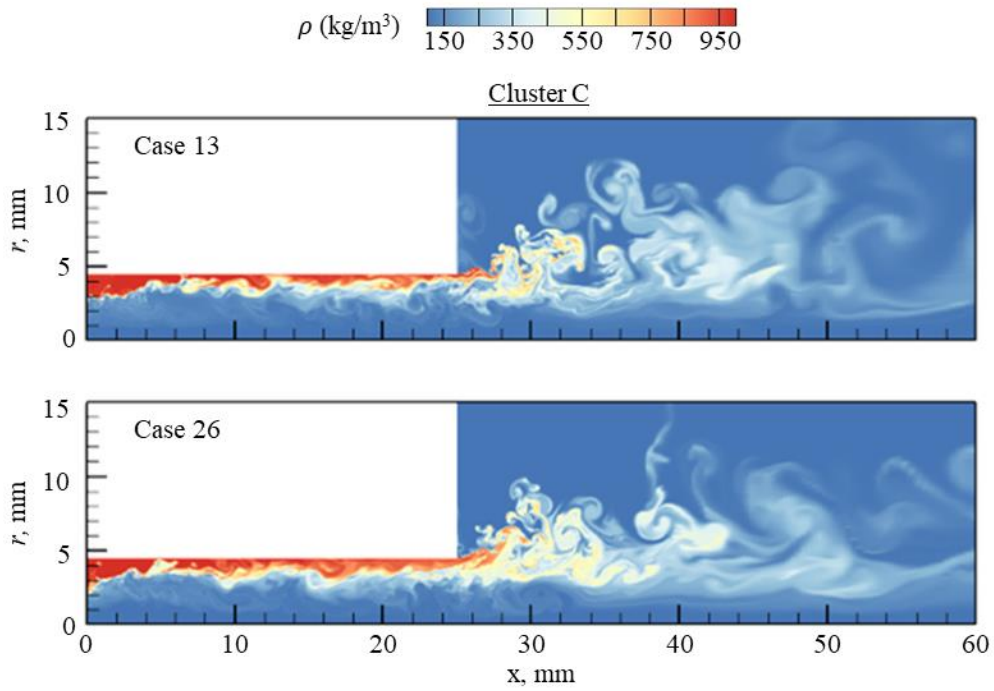

Figure 18—Selected snapshots of density field in Cluster C at *t* = 1.21 ms, obtained from LES-based high-fidelity simulations

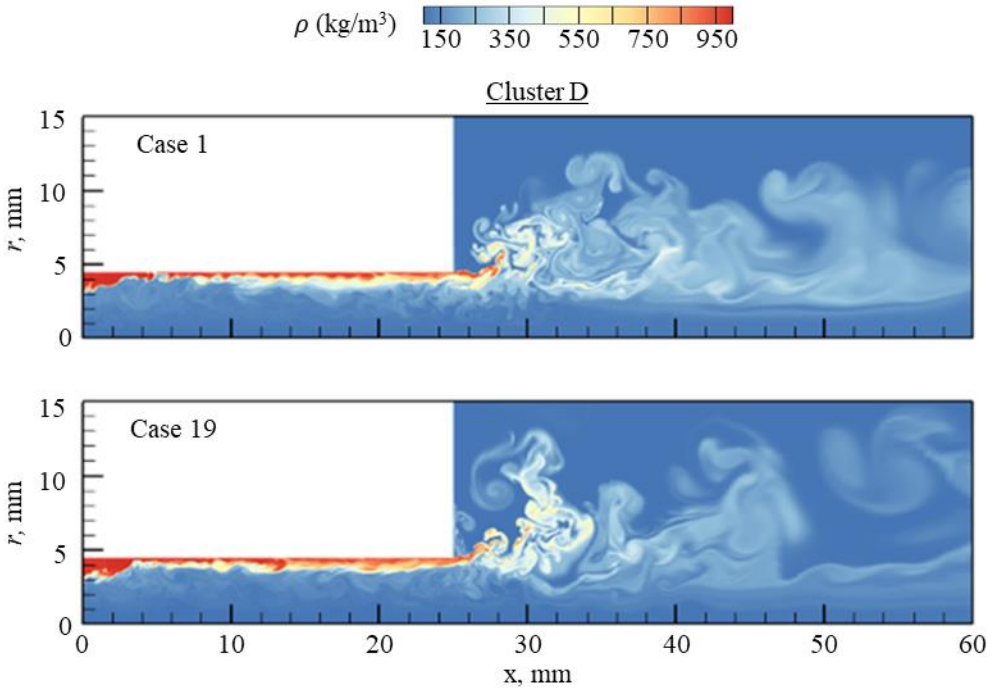

Figure 19—Selected snapshots of density field in Cluster D at *t* = 1.21 ms, obtained from LES-based high-fidelity simulations

4.1.3 Supercritical Combustion Cases

Figure 20 shows a global view of the instantaneous distributions of temperature for Cases 1-7. Cases 1-6 are training cases designed for modeling an emulator. Case 7 is created for validation. All detailed parameters can be found in Tables 4-6. The overall flowfield can be divided into four regimes: propellant injection, flame initialization, flame development, and intensive combustion. Figure 21 presents a schematic of the corresponding locations of these flow regimes. Fuel recess lengths differ among all training and testing cases. Hence, the emulation area (the green shaded area in Figure 21) only utilizes the overlapped downstream area with common-grid rule applied in nonreacting flow. The injection regime primarily consists of center jet and outer swirler, in which GOX and kerosene are injected separately. The flame initiates in the recess region where oxygen and kerosene start the mixing process and is extended to a broader area in the taper region. Intensive combustion is well-distributed in the downstream region for cases with recess region. For Case 6, without recess, however, the interaction of oxygen and kerosene is delayed in the taper region. The combustion zone resides primarily in the downstream upper region close to the injector faceplate, due to a lack of kerosene entrainment to the central jet, which is discussed below. Therefore, the length of the recess affects the profiles of GOX and kerosene injections, and this subsequently determines the characteristics of flame evolution in various regimes. The flow structures in these regimes are analyzed in detail in the following sections.

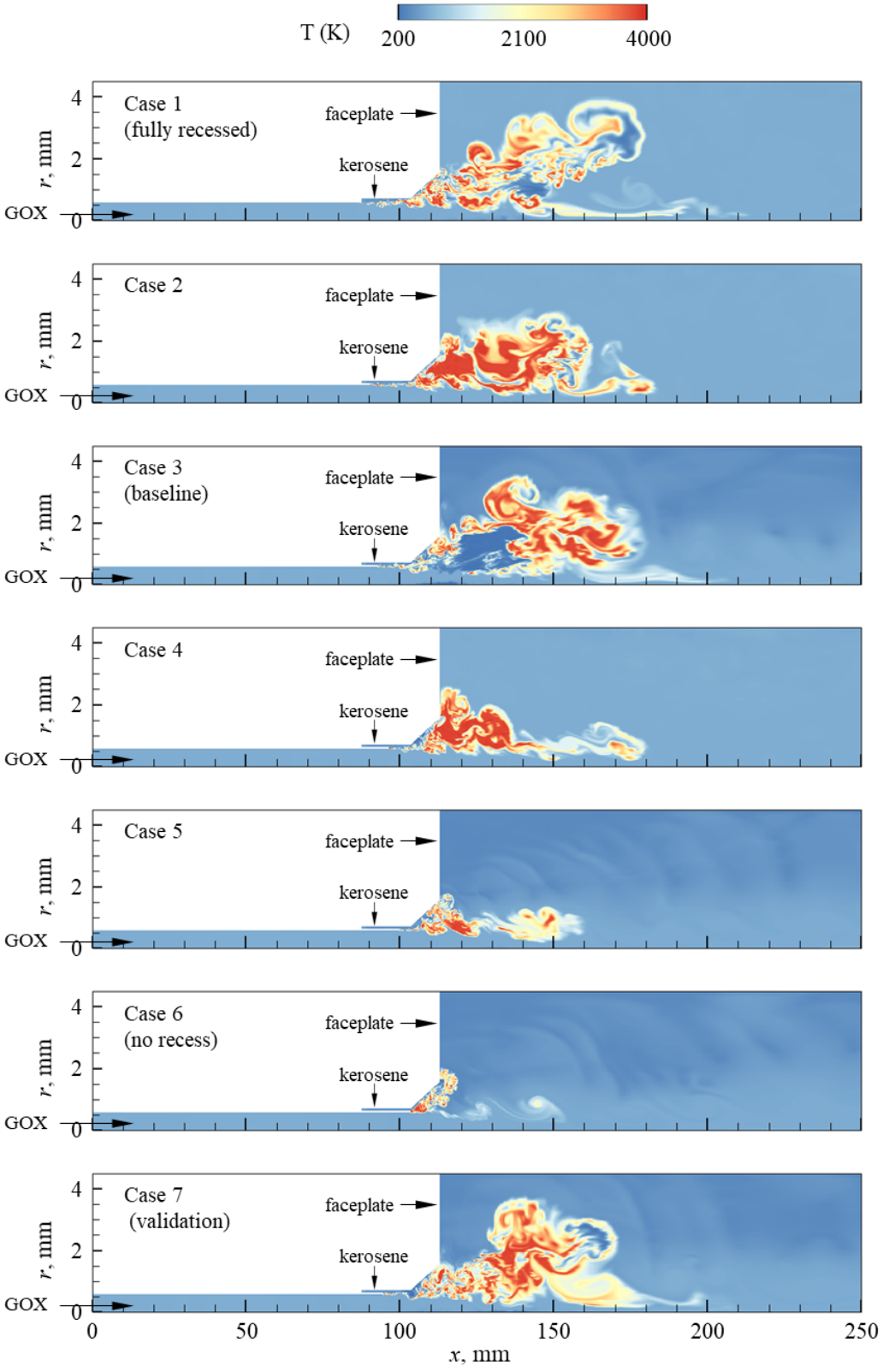

Figure 20—Global view of snapshots of the temperature field for Cases 1, 3, and 6

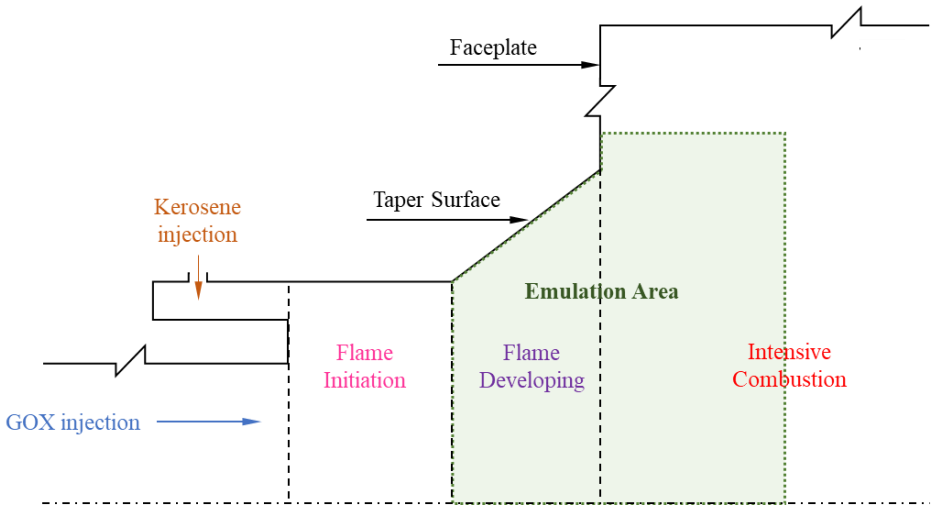

Figure 21—Schematic of flow regimes and emulation area (green shading)

4.2 Common Proper Orthogonal Decomposition (CPOD)

4.2.1 Sensitivity of Injector Geometry Parameters

Liquid-film thickness and spreading angle are two important injector characteristics. An inviscid, incompressible-flow theory predicts the spreading angle as a function solely of the geometric constant [92], and it increases with increasing geometric constant. For real fluids at supercritical conditions, as treated in the present study, the fluid density varies continuously [92]. The spreading angle can be determined based on the slope of the maximum density gradient near the injector exit in a time-averaged sense. As the maximum density gradient is utilized as the boundary for liquid film, the spreading angle and film thickness have variances related to how prominent the maximum density peak appears in the radial direction.

To gage the importance of each injector parameter on the liquid-film thickness and spreading angle, a sensitivity analysis using a Monte Carlo estimate of Sobol indices was performed [75]. Figure 22 shows the primary effects from this analysis. The points indicate the Sobol index estimate for each design parameter, with lines indicating the Monte Carlo integration error for each index estimate. The lines were calculated based on a 95% CI of the estimate. The significant parameters are circled with red solid lines, and attributes that had a minor effect are circled with dash lines. The slot width, δ, is found to be the parameter with the largest Sobol index and thus the strongest influence on the spreading angle. Physically, this can be explained by how geometric parameters govern the inlet flow properties. Assuming a constant mass flow rate, the incoming velocity is inversely proportional to the slot width, and a decrease in slot width increases liquid-film momentum, increasing momentum of the liquid film.

Similarly, the tangential inlet angle (θ) and the slot width significantly affect the liquid-film thickness, while the length (L) and radius (R_n) of the injector have minor effects. The tangential inlet angle controls the direction of momentum. As the injector angle increases, more azimuthal momentum is imparted to the liquid film, thereby increasing the spreading angle at the injector exit. The length and radius can dictate how much viscous loss is experienced by the propellant, as it travels in both the axial and azimuthal directions. The present study, however, finds viscous losses to be a minor effect. Referring to Equation (44), larger values of S_u indicate greater importance of the interaction effect for u. When $|u| = 1$, the sensitivity is called the "mean effect index." Another measure of sensitivity often considered is the "total effect index," which measures the contribution to the output of a given input $\boldsymbol{X}_u$, including all interactions of $\boldsymbol{X}_u$ with other inputs. That is,

$$T_u = S_u + \sum_{l \in u^c} S_{l \cup u} \in [0,1], \tag{64}$$

where u^c is the complementary set of u. Similarly, larger values of T_u indicate greater importance of the effect for u.

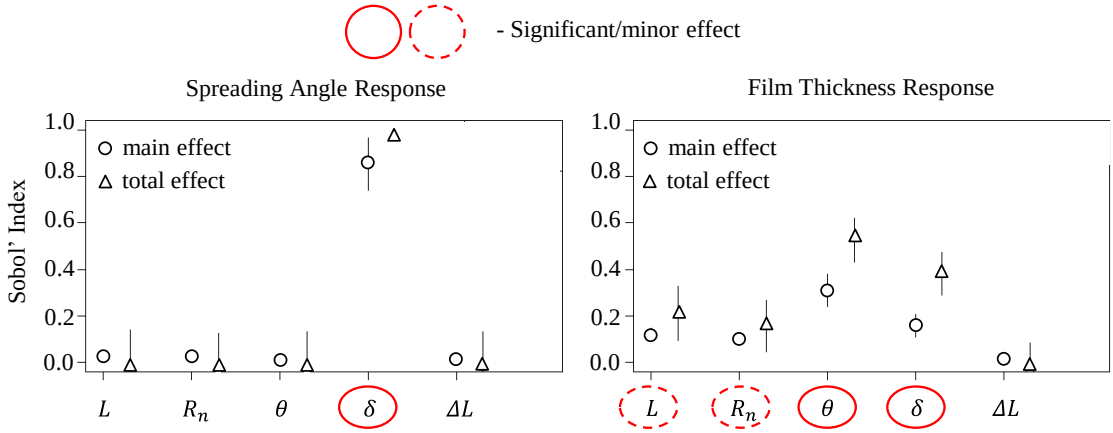

Figure 22—Sensitivity analysis of liquid-film thickness and spreading angle

Figure 23 shows the two-factor interaction effects. It further demonstrates that the main design parameters are the slot width and the tangential inlet angle (with interaction effects circled in blue), which couple to affect the liquid-film response. This is not surprising, as slot width and inlet angle govern flow area and direction of momentum, respectively. The mass and momentum conservation equations are inherently coupled to govern the flowfield.

As previously mentioned, the empirical geometric constant for a swirl injector can be employed to estimate the film thickness and spreading angle, using the hydrodynamics theories described by Bazarov and Yang [92]. These theories, however, assume

incompressible, inviscid flows and can only be used as a preliminary guide. In real injectors, viscous and compressibility effects must be considered. The liquid viscosity results in boundary layer formation along the walls, which causes spatially non-uniform velocity profiles. A primary effect of compressibility lies in the existence of acoustic waves [92]. The supercritical conditions within high-pressure systems make these effects even more pronounced. High-fidelity simulations accounting for real-fluid effects are required to address these issues [12].

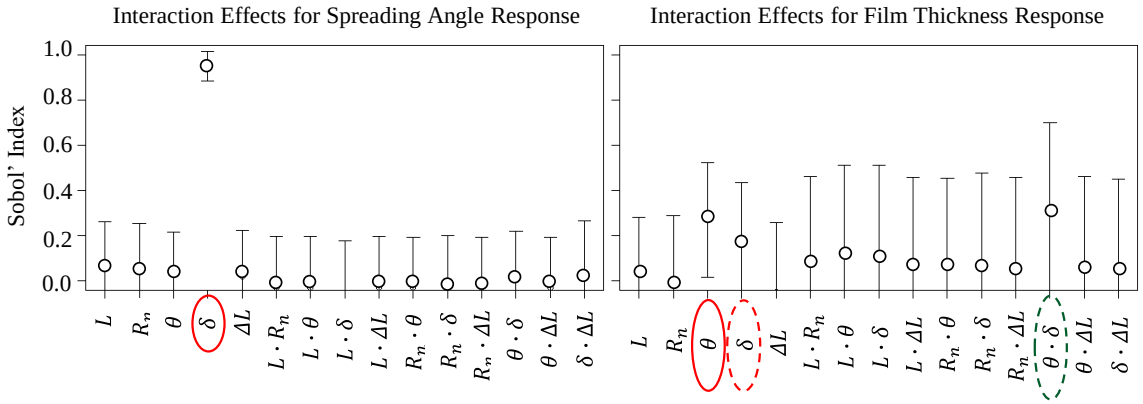

Figure 23—Two-factor interaction of liquid-film thickness and spreading angle

4.2.2 Decision Tree Exploration of Injector Design Space

Further examination of simulated design points shows a clear distinction between two underlying physical phenomena. The first is the expected swirling film that noticeably spreads radially upon exiting the injector. The other is a jet-like behavior of the liquid film where the radial spreading is weak. The DoE methodology utilizes space-filling properties, such that design points in both regimes are simulated. This section explores how to efficiently incorporate this information into the CPOD methodology to refine prediction results.

Designs A and B (whose geometric parameters are listed in Table 8) are each arbitrarily chosen from among the simulated design points as the baseline geometry for determining off-design points. By offsetting injector parameters, two benchmark design points are obtained (denoted as red points in Fig.7). Design A is classified with swirling behavior. Although Design B is classified with jet-like behavior in its developing stage, the flowfield transitions to a swirling flow in its stationary state. This trend may be an indicator that Design B is near the jet-swirl regime boundary. Its stationary state was used to classify this hybrid physics case.

A full design trade-off study requires quantifying how every parameter affects key performance metrics. Hence, all injector variables are retained for the first benchmark, E. The second benchmark, F, only varies design parameters with significant effects on the liquid-film response. The corresponding geometries are shown in Table 10. For Benchmark E, each design parameter deviates +10% from that of Design A. With normalized parameters, the distance traversed in the design space is estimated to be about 18.1%, as calculated in the L_2 linear sense.

Table 10—Injector geometries for benchmark cases

Benchmark	L (mm)	R_n(mm)	θ (°)	δ (mm)	ΔL (mm)
E	22.0	3.22	58.2	0.576	3.42
F	37.7	3.06	59.0	1.417	1.00

The sensitivity study shows that the injector radius and the injection location have less effect than the slot width and tangential inlet angle on the film thickness and spreading angle. They are thus fixed, and the other three parameters are offset from Design B by–

10% to explore the design space at Benchmark F. The closest two simulation points are Designs C and D. The neighboring points are provided because Design B seems to be near the jet-swirl dichotomy.

The second component of the data-driven framework for the design survey is a decision tree [79]. Figure 24 shows the decision-tree splitting process, indicating how the algorithm decides how an injector parameter dictates whether the flow is jet-like or swirling. The initial decision between the two behaviors is achieved by assessing the extent to which the liquid film spreads radially from the injector exit. The numeric outputs are essentially binary flags between the two subgroup classifications. For example, the first numeric output, $\theta < 60.02°$, splits the dataset into 11 jet-like and 19 swirl cases. The decision tree then further classifies the data according to the injector inlet and radius. Intuitively, when the tangential inlet angle, θ, is smaller, there is less azimuthal momentum in the liquid film to cause radial spreading. When the injector inlet, δ, becomes large, the decreased momentum results in jet-like behavior. The decision tree quantifies these effects and predicts a jet-like injector with $\theta < 60.02°$ and $\delta > 1.40$ mm. Following the previous two criteria, if the tangential inlet angle is large enough, that is, $\theta > 49.24°$, the injector retains swirling behavior.

These two benchmark cases are used to verify the decision tree. With such an algorithm, simulation results can be predicted using the model with proper training data. As the next section further details, the emulator relies upon the set of common basis functions extracted from the dataset. With two different types of underlying coherent structures, the two datasets should be trained separately to predict design parameter sets that lead to their corresponding flow behavior.

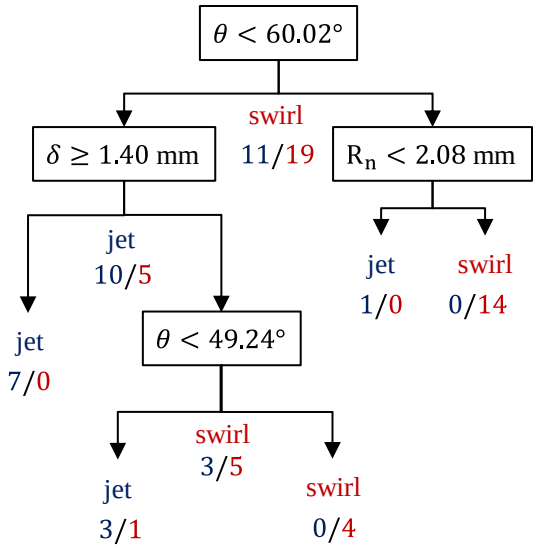

Figure 24—Decision-tree splitting process with numeric classifiers

4.2.3 Emulation Results

To train an emulator and make predictions, a set of common basis functions must be utilized, as previously mentioned. Figure 9 shows the process of generating the common grid. The red lines partition the axisymmetric domain for each case into five regions: injector headend region, injector interior, and three sub-regions downstream of the injector. The densest grid system among the 30 training cases is selected as the common grid, upon which the partitioned regions for all other cases are then scaled to the corresponding regions in the common grid. This scaling is designed such that the ensuing model can leverage common basis functions without significantly changing the flow features of interests. It should be noted the scaling has marginal impact upon liquid-film development visualization within the injector, which has the broadest range among the design parameters. The original data is interpolated with an inverse distance weighting interpolation method using the ten nearest neighborhood points, to retain the fine points in

regions of interest, specifically near the liquid film. The results on the common grid are used for POD analysis.

Figure 25 shows the energy spectrum of the azimuthal velocity captured by the CPOD analysis. This spectrum is chosen as a demonstrative example, as the overall behavior is shared by all other physical variables. Forty-five CPOD modes are required to retain 99% of the energy and limit the corresponding truncation error for the reconstruction. The leading two modes are presented in Figure 26, both indicating swirling flow structures with dominant fluctuations near the injector wall. The flow evolution within the injector and subsequent liquid-film development downstream of the exit are clearly observed.

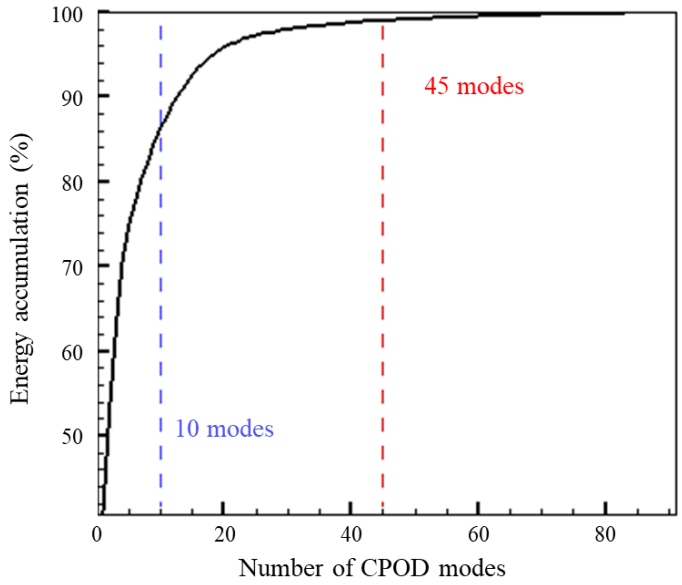

Figure 25—Energy spectrum of CPOD modes for azimuthal velocity component for Benchmark E

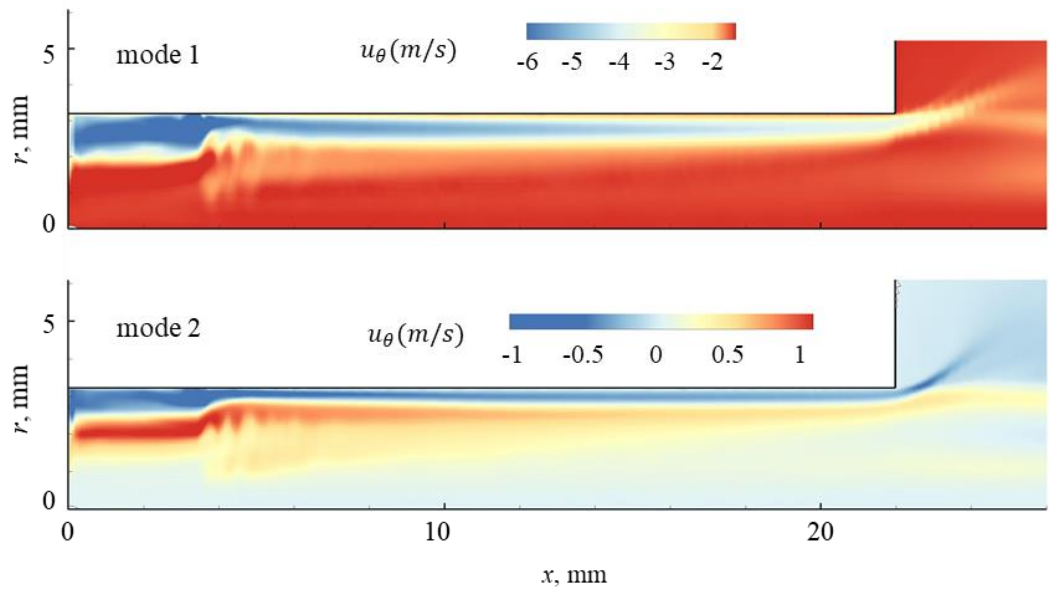

Figure 26—The first two CPOD modes of azimuthal velocity for Benchmark E

The kriging of time-varying coefficients combined with the CPOD modes allows for emulation of the spatiotemporal evolving flow at a new design point. The CPOD modes represent the common physics extracted from the training dataset. A new injector geometry is assumed to produce similar flow physics, including a hollow gas core, a swirling liquid film attached to the wall, and a conical liquid sheet spreading outward at the injector exit. Figure 27 shows snapshots of the temperature field for the simulation and emulations of Benchmark E ($L = 22.0\,mm$, $R_n = 3.22\,mm$, $\theta = 58.2°$, $\delta = 0.576\,mm$, and $\Delta L = 3.42\,mm$). For the temperature CPOD analysis, 2,000 of the 30,000 modes that can be extracted are required to capture 90% of the energy and are used for the prediction. Good agreement is obtained, illustrating the same qualitative trends for the flow structures, with a liquid film along the injector wall and a center recirculating flow downstream of injector. The POD analysis can be interpreted as a spatial averaging technique using the covariance matrix of the flow variable of interest. Some flow details, such as the surface wave

propagation of the liquid film, may be smoothed out due to averaging. This concern can be addressed effectively using the aforementioned statistical and optimization algorithms to tune GP model parameters. The resultant emulator model thus mitigates the smoothing effects and captures the flow structures well.

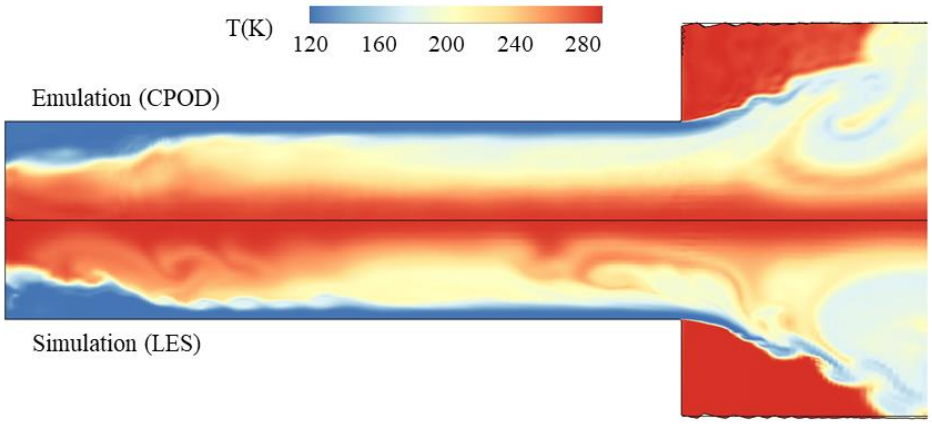

Figure 27—Comparison between simulation and emulation of instantaneous temperature distributions for Benchmark E

4.2.3.1 Response Performance Metrics

As a preliminary comparison, a kriging surrogate model was applied to the extracted liquid-film thickness and spreading angle at the injector exit. The training process was implemented for the 30-case dataset. The following discussion is based on Benchmark E: a swirl case. The liquid-film thickness is estimated, based on hydrodynamic theories, to be 0.618 mm, and the spreading angle 91.8°. The single-point emulator predicts a liquid-film thickness of 0.520 mm and a spreading angle of 99.0°. The data are compared with the simulation results of 0.430 mm and 103°, respectively. Figure 28 shows the variation of the film thickness along the injector wall.

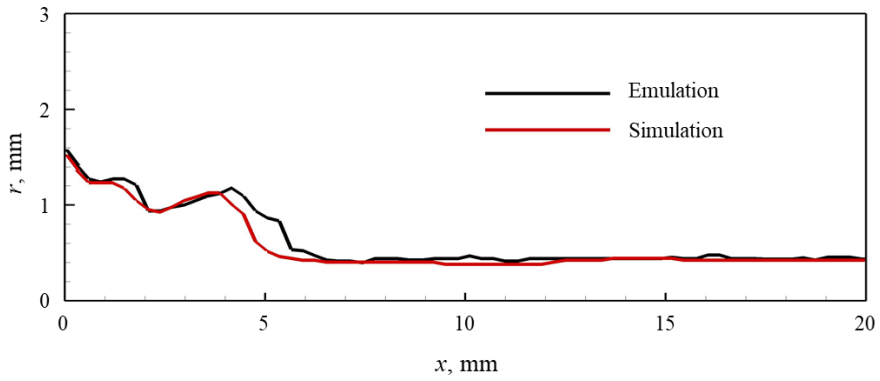

Figure 28—Comparison of mean liquid-film thickness along axial distance

At the injector exit, the time-averaged film thickness and spreading angle predicted by the kriging surrogate model are 0.420 mm and 107°, corresponding to percentage errors of 2.38% and 3.88%, respectively. The model matches the simulation in terms of key features such as the liquid-film distribution and spreading angle, performance measures that are needed for assessing injector design.

For Benchmark F, the baseline case (Design B) develops from jet-like to swirling behavior, as shown in Figure 29. The design parameters are near a critical hyperplane separating different flow features.

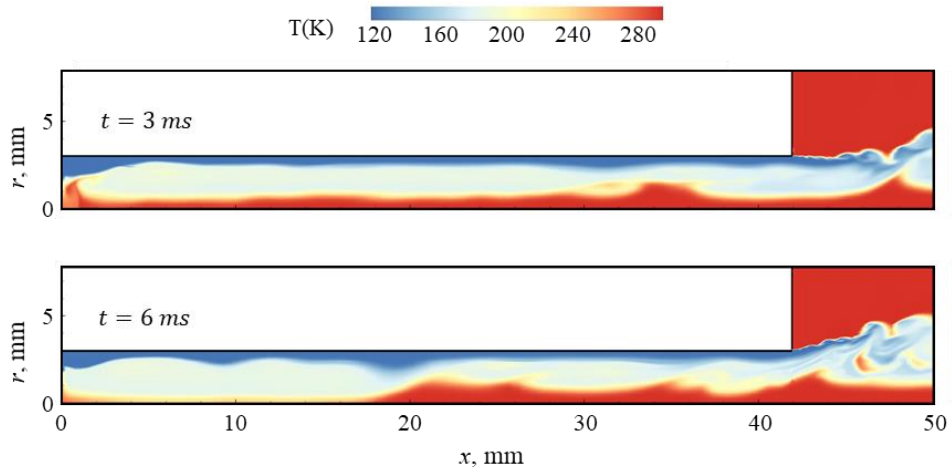

Figure 29—Time evolution of the temperature for baseline case for Benchmark F

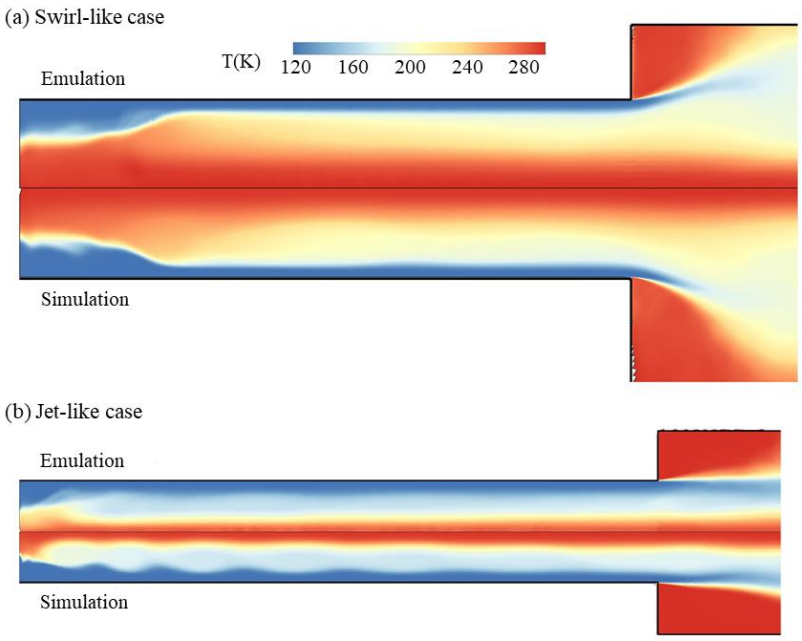

Figure 30—Mean temperature distributions for benchmark cases (a) swirl-like and (b) jet-like

Figure 30 shows the time-mean temperature distributions for the two benchmark cases. The accumulation of liquid propellant at the injector headend is observed in both results. The liquid-film thickness and spreading angle match well. For Benchmark F, which produces a jet-like flow, a standing wave appears in the upstream portion of the injector. The emulation result captures the wavy structure only to some extent. In the downstream region, the liquid-film thickness and spreading angle are better predicted. In the region where the film breaks apart, less propellant appears in the simulation result.

4.2.3.2 Root Mean Square Relative Error (RMSRE)

The root mean square relative error (RMSRE) is defined by

$$RMSRE(t;S) = \frac{[\int_S \{f(\boldsymbol{x},t;\boldsymbol{c}_{new}) - \hat{f}(\boldsymbol{x},t;\boldsymbol{c}_{new})\}^2 d\boldsymbol{x}/|S|\,]^{1/2}}{\max\big(f(\boldsymbol{x},t;\boldsymbol{c}_{new})\big) - \min(f(\boldsymbol{x},t;\boldsymbol{c}_{new}))} \times 100\%, \qquad (65)$$

where S is the desired region, $|S|$ is the number of gridpoints under S, $f(\boldsymbol{x},t;\boldsymbol{c}_{new})$ is the simulated flowfield at geometry $\boldsymbol{c}_{new}$, $\hat{f}(\boldsymbol{x},t;\boldsymbol{c}_{new})$ is the emulated flowfield, and $\max\big(f(\boldsymbol{x},t;\boldsymbol{c}_{new})\big)$ and $\min\big(f(\boldsymbol{x},t;\boldsymbol{c}_{new})\big)$ are the maximum and minimum values of $f(\boldsymbol{x},t;\boldsymbol{c}_{new})$ over $\boldsymbol{x}$, respectively.

Table 11 lists the RMSRE for the two benchmark cases. This quantitatively compares the simulation and emulation shown in Figure 30, illustrating minor discrepancies near the injector wall. For the jet-like case, the error is reduced if only the upstream results (that is, upstream of the injector exit) are considered.

Table 11—RMSRE of temperature distribution

Benchmark	Overall	Upstream	Downstream
E (swirl)	5.18%	6.62%	3.10%
F (jet-like)	8.65%	8.30%	9.03%

Figure 31 shows the time-mean temperature distribution in the radial direction at various axial locations for Benchmark E. The high gradient region represents the transition between the liquid film and the gas core. There is a slight deviation in this transition region, where the simulated temperature gradient is sharper than that of the emulation. Similar results, not shown, are also obtained for Benchmark F.

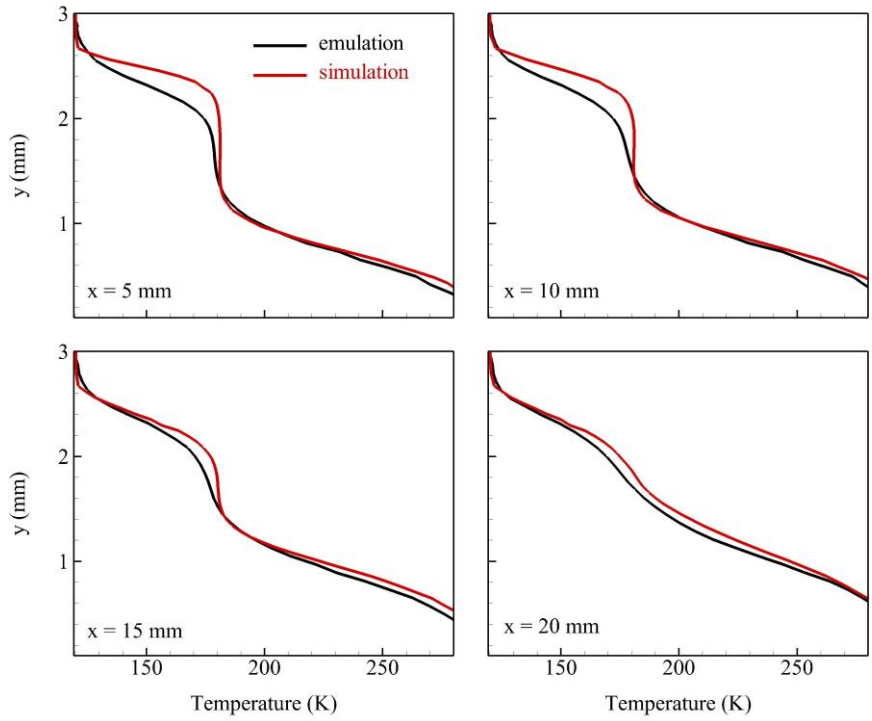

Figure 31—Mean temperature distribution in radial direction for Benchmark E

To illustrate the importance of incorporating the decision tree within the framework, a comparison is made with a prediction from an emulator without dataset classification [1]. Table 12 lists the RMSRE for the two benchmark cases using the emulator trained with the entire dataset. The Benchmark E results are slightly worse, and Benchmark F's prediction is significantly off.

Table 12—RMSRE of temperature distribution results (without dataset classification)

Benchmark Case	Overall	Upstream	Downstream
E (swirl)	5.93%	6.70%	5.09%
F (jet)	13.2%	7.43%	17.7%

Next, axial velocity is used as training data, demonstrating the capability of modeling other flowfield variables. Figure 32 shows the time-mean distribution comparison between the simulation and emulation for Benchmark E. The key flow features, such as the gaseous core and swirling film, are predicted well. The RMSRE listed in Table 13 numerically outperforms temperature results. This improvement can be explained by the broader range, which leads to higher errors.

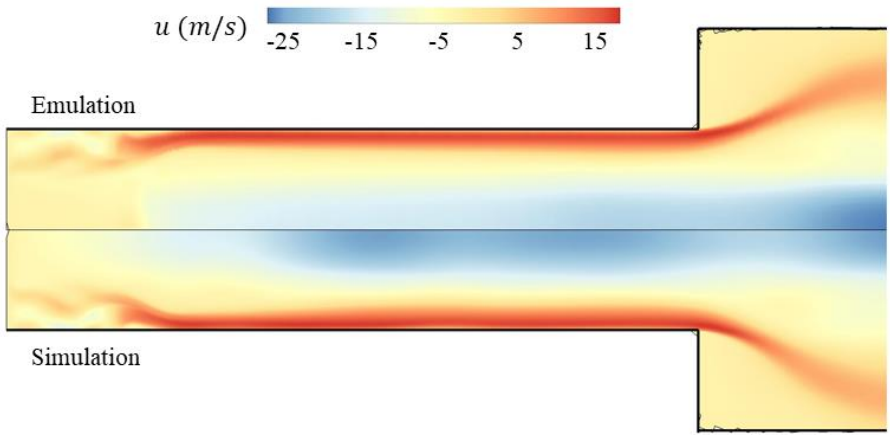

Figure 32—Mean axial velocity distribution for Benchmark E

Table 13—RMSRE velocity distribution results

Benchmark	Overall	Upstream	Downstream
E (swirl)	4.12%	4.58%	3.64%
F (jet)	3.97%	4.71%	2.85%

Figure 33 shows the time-mean axial velocity distribution in the radial direction for various axial locations in the injector for Benchmark E. The transition region is matched, although there is a deviation near the injector centerline, where the gradient is smoother in the emulator prediction. Similar results, not shown, were seen for Benchmark F.

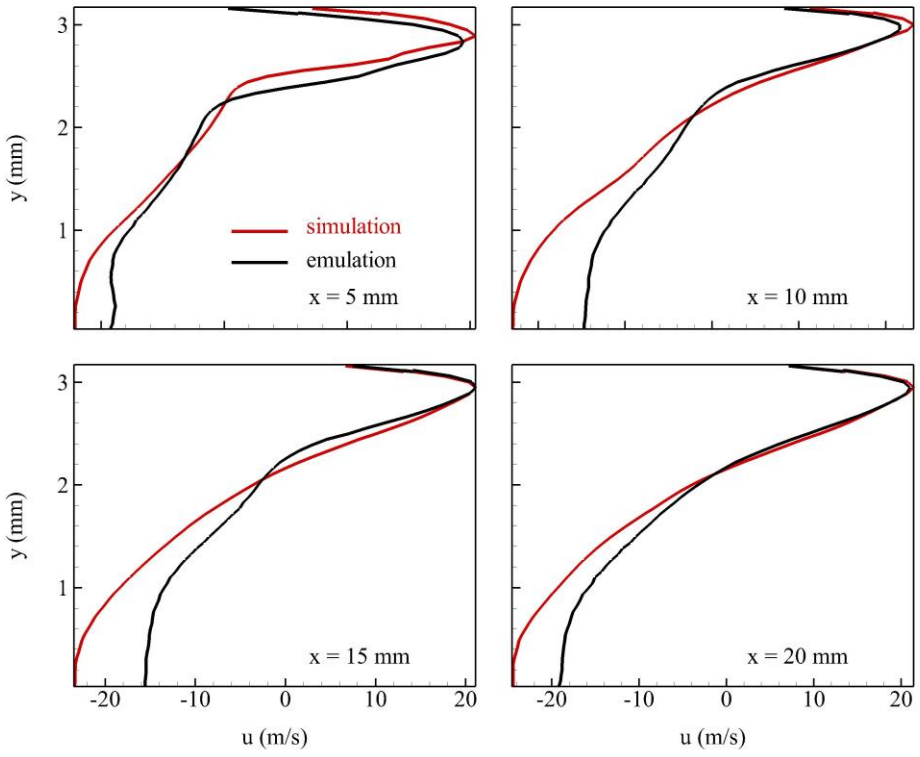

Figure 33—Mean axial variation of velocity distribution in the radial direction for Benchmark E

4.2.3.3 Injector Dynamics

Injector dynamics involve downstream pressure fluctuations, causing pressure drop oscillations across the liquid film. These changes in turn trigger mass flow rate variations across the tangential inlets [92], over a wide range of time scales. A power spectral density (PSD) analysis can quantify these oscillations and capture the periodicity of flow features. Mathematically, the PSD can be interpreted as the Fourier transformation of the autocorrelation function for a signal.

Figure 34 shows the position of pressure probes in the fluid transition region. The probes are located near the film surface to measure pressure fluctuations. The pressure

PSDs are calculated for both the simulation and emulation results. Figure 35 shows the PSD of Probes 1, 3, 5, and 7; the frequency content is observed to be well quantified.

The high-frequency oscillations that are typically present in swirl injectors with a vortex chamber are not prominent. Most of the signal is comprised of low-frequency content, representing surface wave propagation along the film. In addition, acoustic waves propagate, couple, and interact with hydrodynamic waves, appearing as several different frequencies. The simulated and emulated probes show similar dynamics, such that the peak frequencies of the simulation and emulation results match. However, the emulator amplifies the dominant frequencies, as the kriging model may be overfitting slightly due to insufficient data. Despite this signal strengthening phenomenon, the analysis displays an ability to model flow dynamics, properly capturing the simulated periodic oscillations. Downstream of the injector exit, the dynamics become more broadband and no dominant oscillations appear, because there exist strong interactions between the shear layer and recirculation zone generated from vortex breakdown.

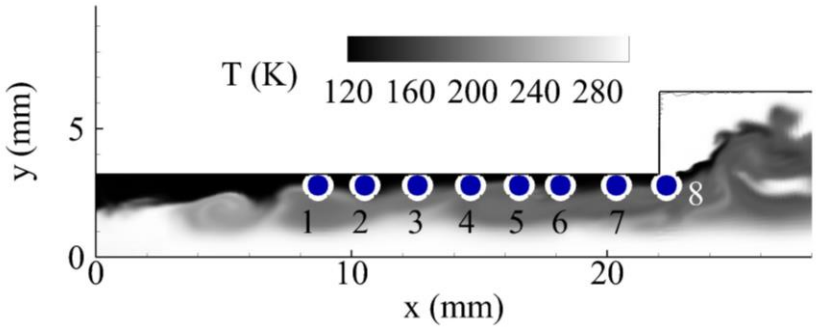

Figure 34—Probe positions along the liquid-film surface

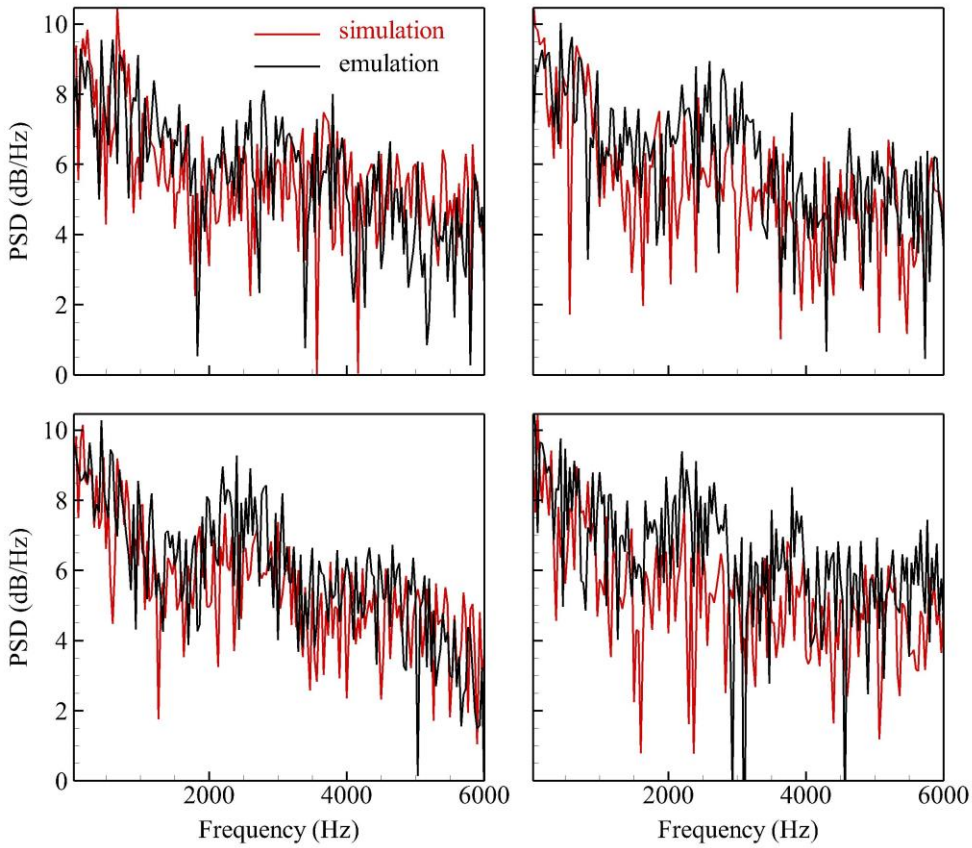

Figure 35—PSD results of pressure fluctuations for Probes 1, 3, 5, and 7

4.2.3.4 Uncertainty Quantification (UQ)

In addition to the aforementioned validation methods, the emulator model allows for quantification of predictive uncertainty, which can be used to define CIs for model fit. Moreover, these uncertainties can be linked to dynamic flow physics. As an example, the spatial UQ is shown in Figure 36, displaying the one-sided width of the 80% CI for pressure and temperature (a derivation of this interval is found in [1]). The uncertain areas, in the time-mean temperature distribution, correspond to the most dynamic sections of the liquid transition region. The downstream uncertainty is caused by the recirculation induced through vortex breakdown. Further details of UQ analysis are discussed in Section 4.5.

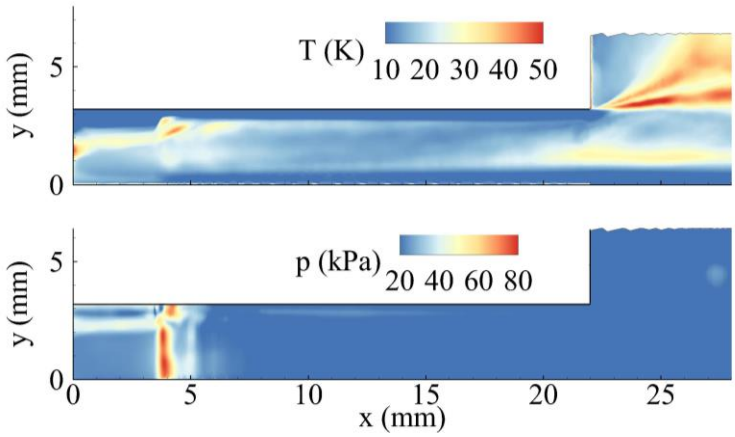

Figure 36—One-sided width of the 80% CI for Benchmark E: temperature and pressure predictions

4.2.3.5 Computation Time

Figure 37 presents the simulation and emulation timeline. Computation times are calculated based on performance for a parallelized system of 200 Intel Xeon E5-2603 1.80 GHz processors. A total of 900,000 CPU hours is required for the 30 GB dataset. The CPOD extraction and parameter estimation for the model takes about 45 minutes.

The parallelized predictions from the developed model only require approximately 30 CPU hours, significantly reducing the turnaround time, as compared with LES simulations requiring 30,000 CPU hours. This improved computational efficiency is crucial, as it enables quick design iterations. The existing spatiotemporal emulators mentioned in the introduction require much more computation time to fit the underlying statistical model, because the training dataset of each simulation is too large to directly manipulate [84]. By carefully using physical knowledge to make informed model assumptions, state-of-the-art machine-learning techniques have been leveraged to develop a methodology offering and efficient strategy to survey the design space.

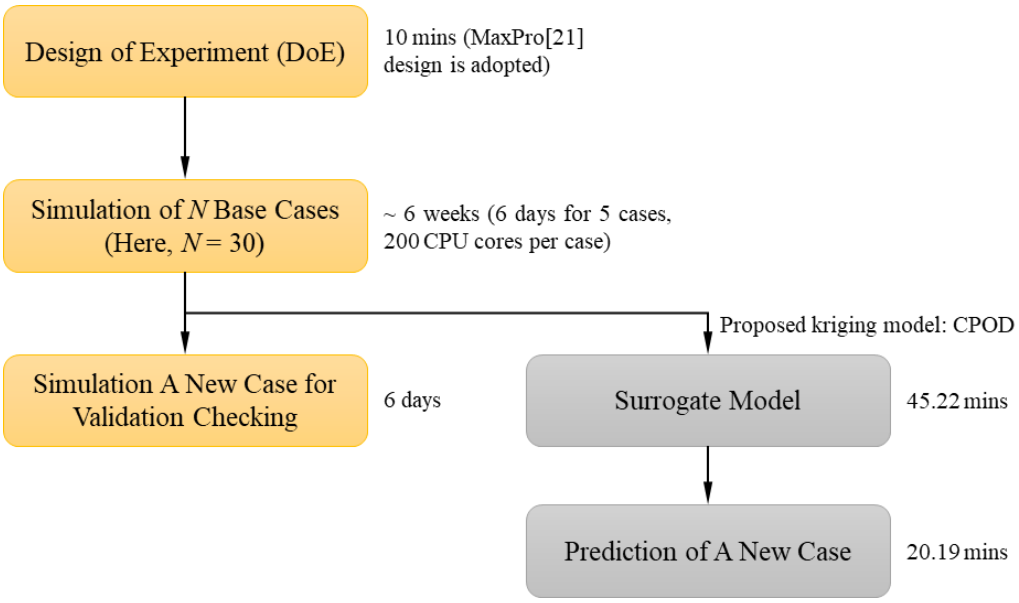

Figure 37—Simulation and emulation timeline

4.3 Kernel-Smoothed Proper Orthogonal Decomposition (KSPOD)

Figure 38 compares the first three pressure POD modes between Cases 9 and 21 in Cluster B. Mode 1 is an ensemble of the longitudinal modes of the hydrodynamic instability. In both cases, the injector length is fixed at 25 mm, and the longitudinal hydrodynamic wave speed in the liquid film is estimated to be 7-9 m/s [10-12, 73]. This leads to a characteristic frequency of 0.32-0.45 kHz for the hydrodynamic instability. Mode 2 contains about 12% of the total energy and has similar structures similar Cases 9 and 21, except in the downstream region close to the centerline. Mode 3 has around 5% of the total energy, and the distributions for the two cases are alike. The dominant frequencies for Mode 2 in the two cases are 0.61 and 0.63 kHz, respectively. For Mode 3, dominant frequencies of 1.14 and 1.39 kHz are observed. The frequencies associated with the

recirculating flow downstream of the injector exit are 0.65 and 1.29 kHz, respectively. Hence, Modes 2 and 3 can be attributed to excitation by the processing vortex core. In short, Modes 1-3 capture similar physical characteristics in the dominant POD modes with the same order resulting from the eigen-decomposition. The similarity of POD modes among cases in the same cluster justifies the two fundamental assumptions of KSPOD stated on page 3.

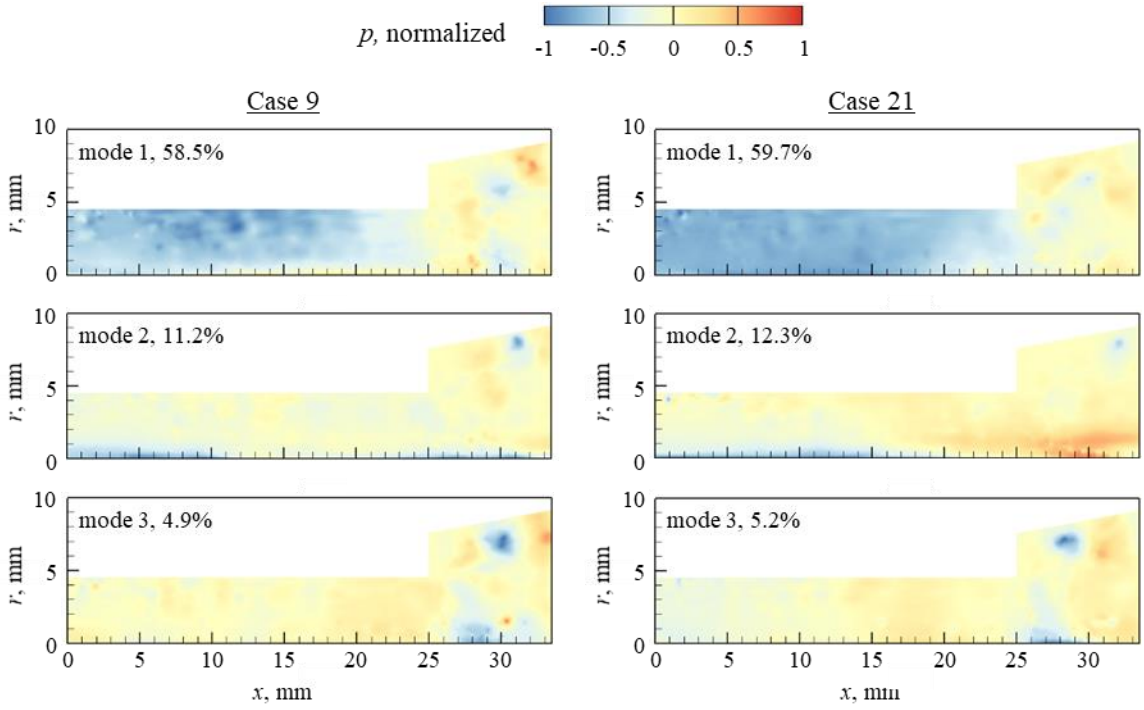

Figure 38—Pressure in POD Modes 1, 2, and 3 for Cases 9 and 21 from Cluster B.

Figure 39 shows the accumulated energy percentage of the density POD modes for the cases presented in Figure 38. The first 150 modes contain over 90% of the total energy, and the first 300 modes have over 99.9% of the energy. The process of KSPOD builds a posterior model based on the basis functions ranked by the eigen-decomposition. As such, the reduced data sets (that is, the dominant modes) capture the significant flow structures and their dynamic characteristics for the same rank.

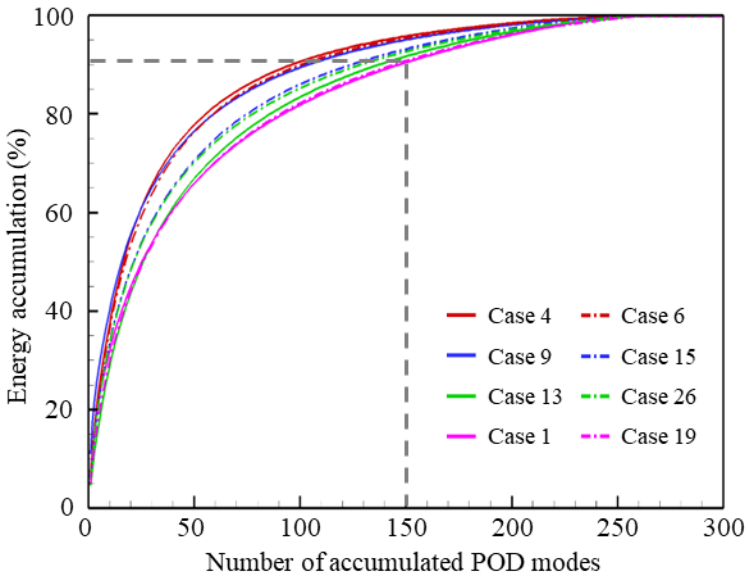

Figure 39—Energy accumulation of density of POD modes

The 30 cases selected by DoE have different inlet velocities, which generate different instability wave speeds. Phase information, however, cannot be reproduced perfectly, due to the presence of turbulence. If two designs with similar dynamic mechanisms, such as Cases 4 and 6, have phase differences of 90° or 180°, the prediction results may be excessively smoothed by traditional kriging. If the weighting numbers evaluated by kriging for these two cases are close and their instability waves are exactly out of phase, the wave information cancels out. As such, KSPOD applies a weighting function for the POD modes to ensure that similar POD modes in different cases can retain appropriate phase information.

Figure 40 shows excellent comparison between the simulation result and prediction from the trained emulator for Test Case A1. The design parameters are given in Table 14. The KSPOD-based emulator works well; it can capture essential flow structures successfully. The evolution of the liquid film and its spreading downstream of the injector

exit also agree well between the simulation and emulation. The overall turnaround time for emulation prediction, excluding data loading and training, is about 42 seconds of CPU time for one snapshot.

Table 15 lists the contribution of POD modes from each case. The weighting number is calculated based on the Test Case A1. It is noted that Cases 17 and 30 respectively provide 91.77% and 13.37% weighting, while Cases 5, 10, 22, and 28 provide over 10% negative weights on A1.

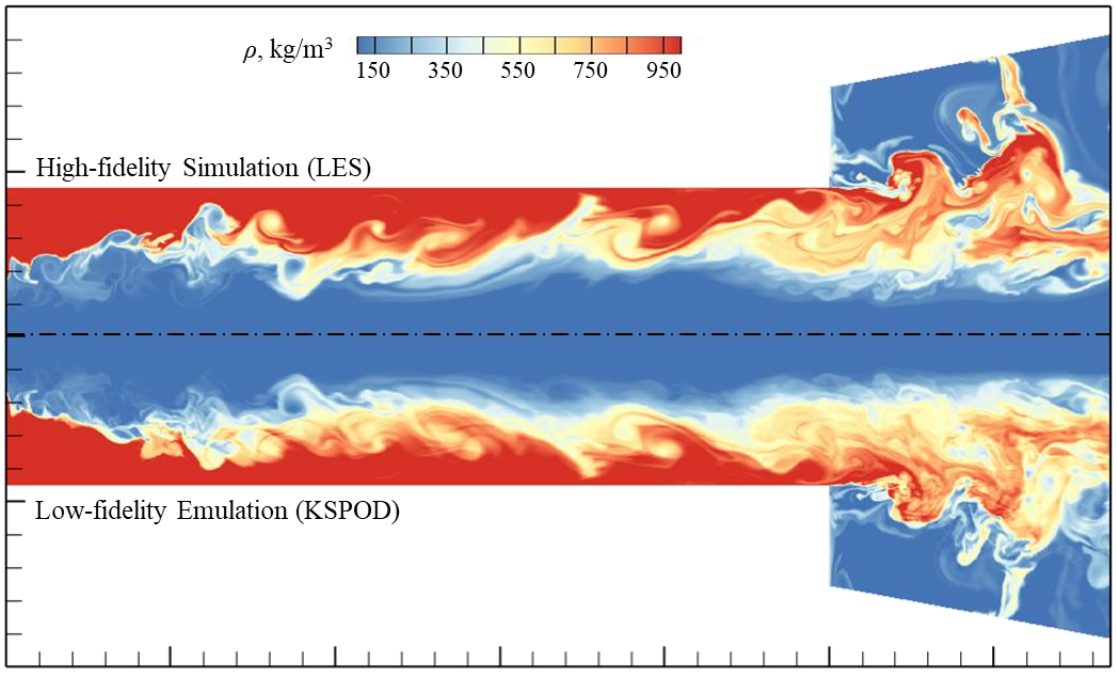

Figure 40—Comparison of density field between LES-based simulation and prediction by KSPOD-based emulation: Test Case A1

Table 14—Design parameters for eight test cases in four different clusters

Case	δ (mm)	θ (deg)	ΔL(mm)	u_{in} (m/s)	u_r (m/s)	u_θ (m/s)	Cluster
A1	1.26	44.11	0.94	6.55	4.70	4.56	A
A2	1.20	41.97	0.90	6.65	4.94	4.44	A
B1	0.70	40.73	2.71	11.12	8.43	7.26	B
B2	0.71	52.59	3.24	13.79	8.38	10.95	B
C1	0.42	37.73	2.41	17.91	14.16	10.96	C
C2	0.49	57.12	2.88	22.33	12.12	18.75	C
D1	0.27	50.39	1.40	34.37	21.91	26.48	D
D2	0.33	60.76	2.32	36.32	17.74	31.70	D

Table 15—Weighting numbers from POD modes for Case A1

Case	1	2	3	4	5	6	7	8	9	10
Cluster	D	B	B	A	A	A	C	C	B	A
Weighting Number	-0.01%	0.50%	2.13%	0.82%	-4.24%	0.02%	0.00%	0.12%	0.04%	-2.18%
Case	11	12	13	14	15	16	17	18	19	20
Cluster	A	A	C	B	B	A	A	A	D	B
Weighting Number	2.17%	-0.03%	0.16%	0.18%	-0.13%	0.00%	91.77%	-0.17%	-0.04%	-0.08%
Case	21	22	23	24	25	26	27	28	29	30
Cluster	B	A	A	A	C	C	B	A	A	A
Weighting Number	0.16%	-2.50%	0.20%	-0.13%	0.05%	0.01%	0.00%	-2.28%	0.08%	13.37%

Figure 41 shows prediction results for the four different design clusters, in each of which two test cases are considered. The flow structures and dynamics are well captured by the emulator. In Case C1, the corner recirculation near the headend of the injector is clearly observed. The discrepancy caused by time delay is present in Cases A1 and B2. In A1, the traveling surface wave in the injector propagates downstream slightly faster in the simulation than in the emulation.

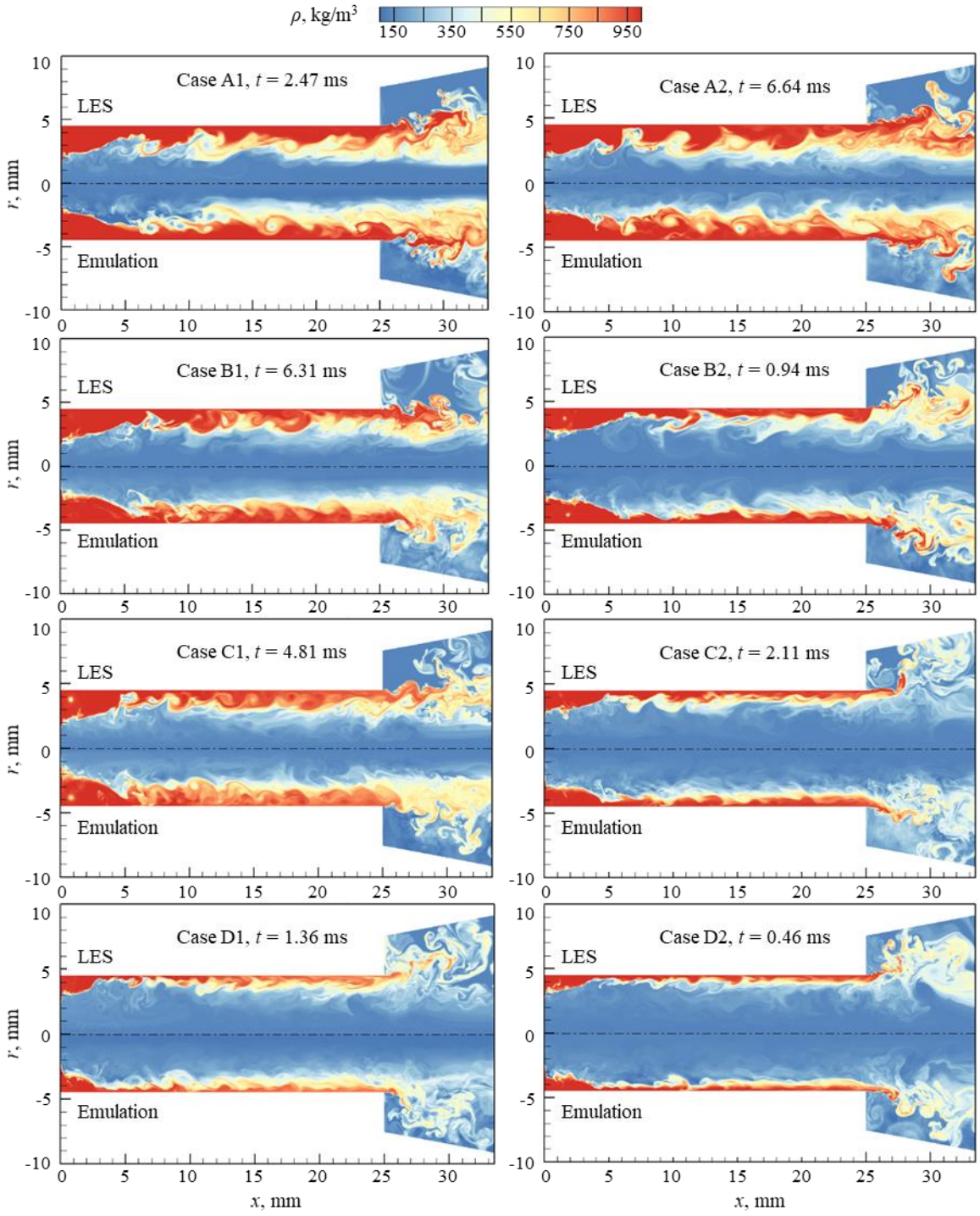

Figure 41—Comparison of density fields between LES-based simulation and prediction by KSPOD-based emulation

4.3.1 Accuracy of Prediction: Film Thickness and Spreading Angle

To further evaluate the accuracy of the KSPOD-based emulation, two performance measures, film thickness and spreading angle, are assessed. Table 16 presents a comparison of the time-mean simulation and emulation (prediction) results, obtained by averaging the instantaneous data over a statistically meaningful time duration. The error is calculated as follows:

$$\varepsilon_{abs} = \frac{|x_{sim} - x_{emu}|}{x_{sim}} \times 100\%, \tag{66}$$

where x_{sim} represents data from simulation and x_{emu} from emulation. As most training cases are located in Clusters A and B, the predictions are more accurate within these two clusters.

Table 16—Film thickness and spreading angle for simulation and emulation results

	Spreading Angle			Film Thickness		
	Simulation	Emulation	$\overline{\varepsilon_{abs}}$	Simulation	Emulation	$\overline{\varepsilon_{abs}}$
Case A1	52.85	52.92	0.14%	0.629	0.625	0.51%
Case A2	52.57	51.96	1.15%	0.637	0.657	3.14%
Case B1	54.22	53.66	1.02%	0.582	0.600	3.03%
Case B2	53.81	53.87	0.12%	0.594	0.592	0.40%
Case C1	57.68	57.71	0.05%	0.469	0.473	0.85%
Case C2	57.78	57.74	0.06%	0.475	0.472	0.63%
Case D1	59.00	58.03	1.64%	0.379	0.378	0.26%
Case D2	61.59	61.33	0.41%	0.370	0.377	1.97%

The probability densities of instantaneous spreading angle and liquid-film thickness for four selected training cases in Cluster A are obtained from the estimated kernel smoothing function, as shown in Figure 42. A kernel distribution is a nonparametric representation of the probability density function, $f_h(x)$, of a random variable, written as

$$f_h(x) = \frac{1}{nh}\sum_{i=1}^{n} \mathcal{K}\left(\frac{x - x_i}{h}\right), \tag{67}$$

where n is the sample size, $\mathcal{K}(\cdot)$ is the density smoothing function, and h a is the smoothing parameter, bandwidth. The distributions of probability density for cases in the same cluster bear close similarities, but do not collapse.

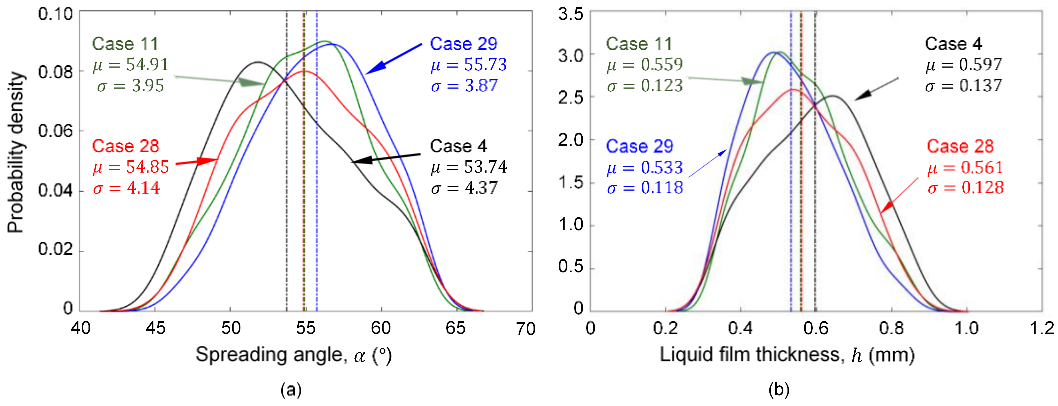

Figure 42—Probability densities of instantaneous spreading angle and liquid-film thickness for Cases 4, 11, 28, and 29 from Cluster A (vertical lines represent mean values)

Figure 43 compares the probability density distributions of spreading angle and liquid-film thickness between simulations and emulations for all test cases. The vertical lines represent mean values. Detailed information about mean values, standard deviations, and averaged absolute error $\overline{\varepsilon_{abs}}$ is given in **Table 16**. The maximum absolute errors for liquid-film thickness and spreading angle are 1.64% in Case D1 and 3.14% in Case A2,

respectively. The liquid-film thicknesses are relatively small in Clusters C and D, so their mean values visually overlap each other.

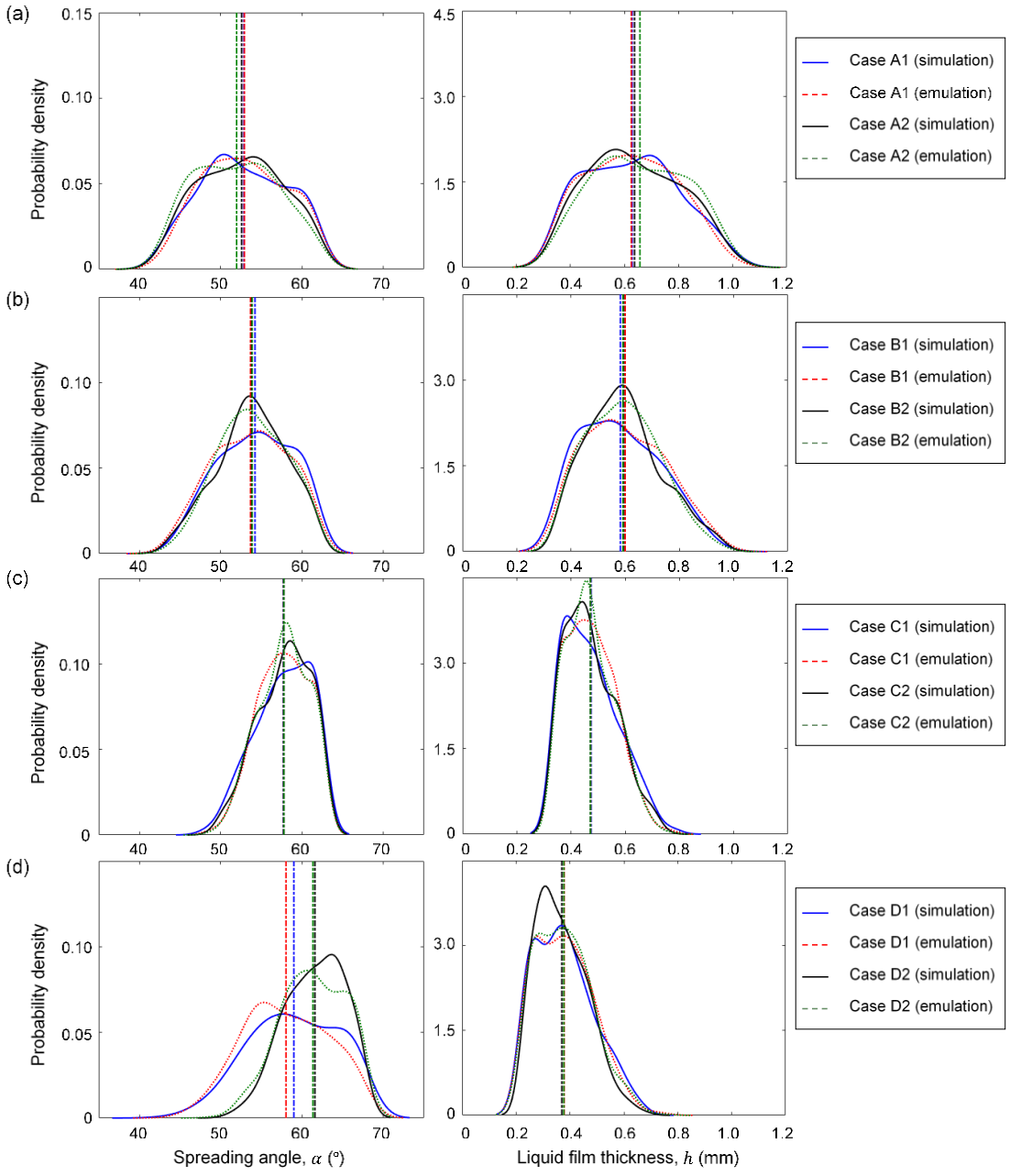

Figure 43—Probability densities of instantaneous spreading angle and liquid-film thickness for test cases (vertical lines represent mean values)

Another method for measuring the performance of the emulation is based on the distribution of liquid-film thickness along the axial direction. Figure 44 shows a comparison between the simulation and emulation results, averaged over 1,000 snapshots. Figure 45 shows the absolute error between simulations and emulations. The horizontal lines represent averaged absolute error $\overline{\varepsilon_{abs}}$ for each test case. Overall, the liquid-film thickness predicted by the KSPOD-based emulation has an averaged error less than 5%, except for Case B2, in which $\overline{\varepsilon_{abs}} = 6.4\%$. Cases with higher inlet velocities have less variation for the liquid-film thickness near the injector exit. The first local maximum of error occurs in the LOX inlet area, the region that contains the highest momentum and kinetic energy. The second peak of error occurs when the flowfield is still developing. The large error along the axial direction in Case B2 can be attributed to the fact that this case is close to the boundary between Clusters B and C and includes more flow mechanisms that are prominent in the cases in Cluster C. The 30 cases used for model training are mostly distributed in Clusters A and B. If the 30 training cases had space-filling properties with respect to inlet velocities, the prediction error could be decreased and the quantitative analysis for Test Cases B2, C1, and C2 could be improved.

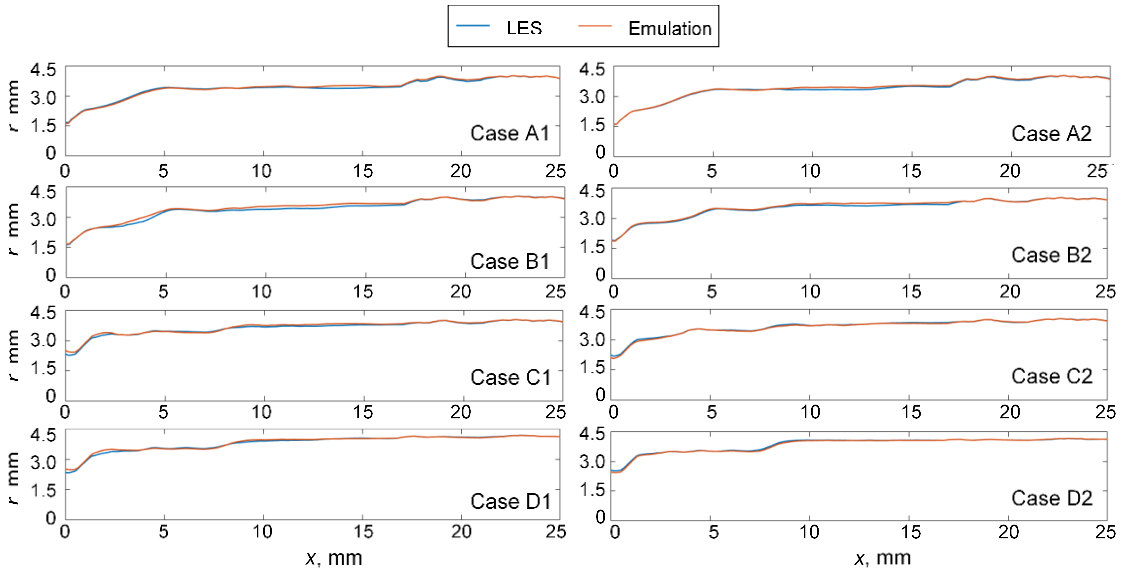

Figure 44—Comparison of liquid-film thickness along the axial direction, averaged over 1,000 snapshots

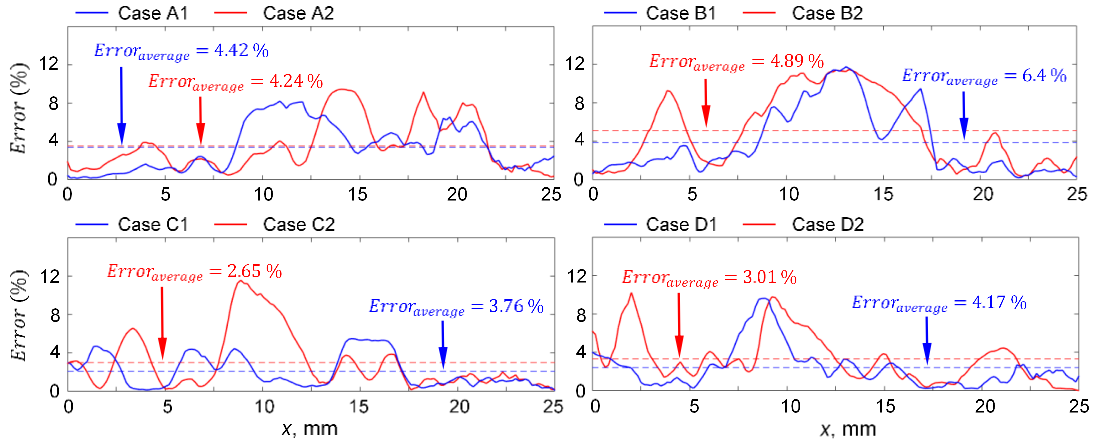

Figure 45—Error for liquid-film thickness along the axial direction

4.3.2 Accuracy of Prediction: Proper Orthogonal Decomposition (POD) and Frequency Comparison

The injector dynamics involved in this example problem cover a wide range of time and length scales. This rich set of physics is quantified using a PSD analysis on a POD coefficient. Injector dynamics involve downstream pressure oscillations causing pressure

drop oscillations across the liquid film, which in turns trigger mass flow rate fluctuations., A spectral analysis, PSD, can quantify these oscillations and capture the periodicity of these flow processes. Mathematically, the PSD can be interpreted as the Fourier transformation of the autocorrelation function of the signal. The PSDs of pressure fluctuations were calculated and compared between the LES and emulation results. Figure 46 shows the comparison between LES and emulation for POD results based on Modes 1 through 4 from Case C2. Here, the first four modes contain over 80% of the total energy.

In Figure 46, the color shapes between simulation and emulation of Modes 1 and 2 are alike, except the downstream parts, where simulation reflects the pressure instabilities more significantly. The color shapes between simulation and emulation of Mode 3 are almost the same, except simulation demonstrates larger magnitude in the flow intake area. The differences between simulation and emulation become more obvious in Mode 4. Compared with simulation, the increasing distribution, with a minimum at the head and an enlarged value at the exit, is stronger in emulation.

In addition to the similarities in POD modes, emulation performs well in capturing dominant frequencies in PSD analysis. From Modes 1 to 4, the trends of all PSD curves are similar between simulation and emulation. Moreover, emulation captures the dominant frequencies identically to simulation in Mode 1 (0.49 kHz) and Mode 2 (0.65 kHz), which occupy over 73% of the total energy. Although emulation dislocates the dominant frequency (0.23 kHz) in Mode 3, its secondary dominant frequency still parallels the dominant one in simulation (0.65 kHz). Consequently, the emulation results based on KSPOD can not only predict flow dynamics with visually pleasant results but also

accurately capture flow physics, including liquid-film thickness, spreading angle, and POD analysis results.

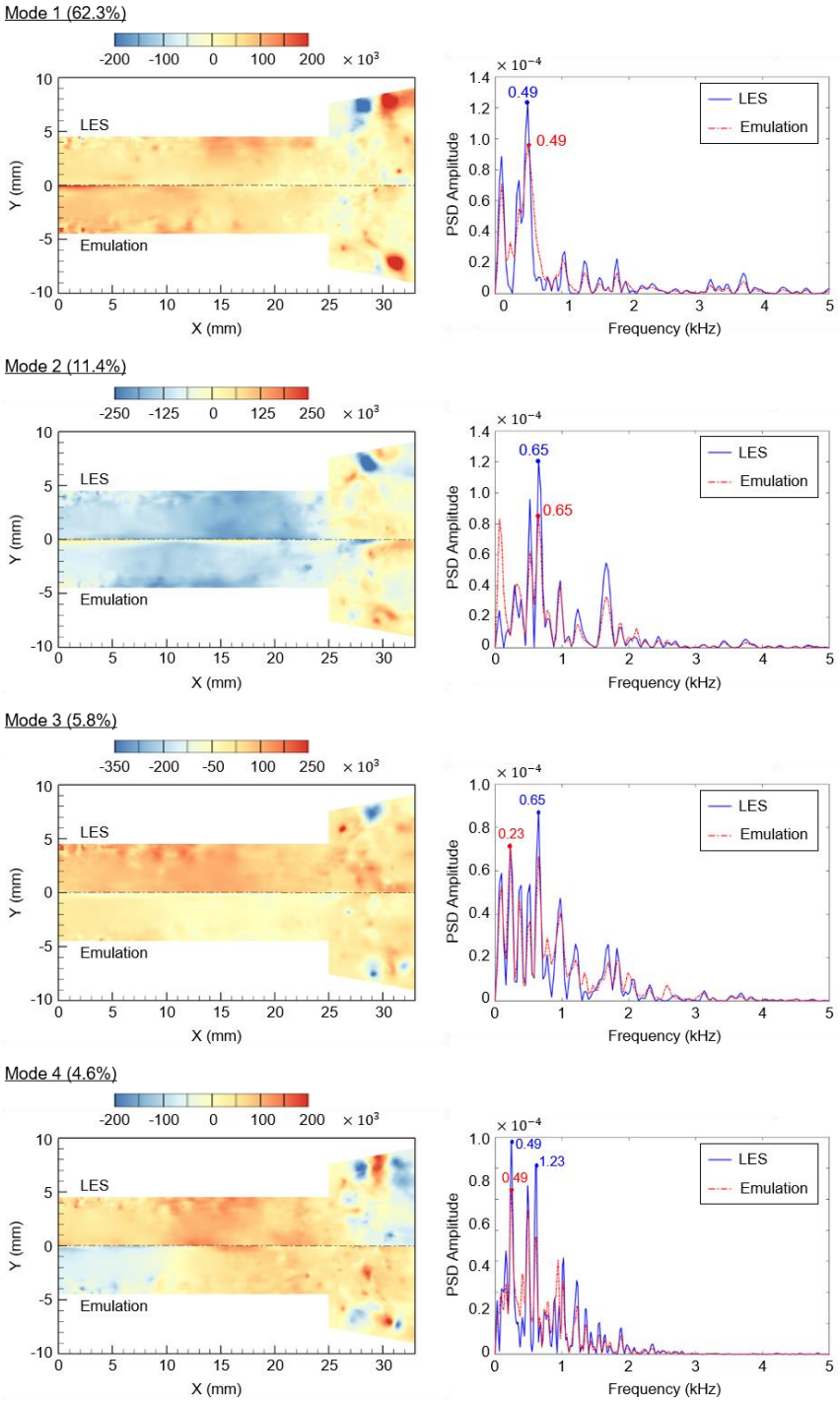

Figure 46—Pressure POD Modes 1-4 and corresponding power spectrum densities for density based on the C2 design

4.4 Common Kernel-Smoothed Proper Orthogonal Decomposition (CKSPOD)

4.4.1 Modes from CKSPOD

The 30 cases applied in this work are the same as those used in our previous KSPOD work [3]. All training cases are obtained by the LES-based high-fidelity numerical framework described earlier and selected by SLHDs for the three design parameters, which are decided by the sensitivity analysis using a first-order Monte Carlo estimation of Sobol indices [2]. The sensitivity analysis was performed based on the desired liquid-film thickness and spreading angle. The three chosen parameters are significantly determined by the inlet velocity, u_{in}, which ranges from 5.71 to 40.43 m/s, as listed in Table 14 [3]. In the KSPOD work, the 30 training cases are roughly categorized into four groups in terms of u_{in} (m/s), as follows: $u_{in} < 10$ for Cluster A; $10 \leq u_{in} < 18$ for Cluster B; $18 \leq u_{in} < 25$ for Cluster C; and $u_{in} > 25$ for Cluster D. Overall, there are 15 cases in Cluster A, 8 in Cluster B, 5 in Cluster C, and 2 in Cluster D. To validate the accuracy of a new surrogate model, each cluster is assigned with two test cases. The design parameters for these test cases are given in Table 14.

The geometric constant, *K*, is an indicator of swirl strength that plays a pivotal role, with a slight change causing significant differences in flow dynamics. The azimuthal momentum, which affects the size of the spreading angle of the liquid film, increases and decreases with *K*. Hence, the training process must preserve fundamental flow physics during data reduction.

In Chang et. al. (2018) [3], the 30 cases selected by DoE have different inlet velocities, which generate different instability wave speeds. Phase information, however,

cannot be reproduced perfectly, due to the presence of turbulence. In KSPOD research, it is proven that a phase difference of 90° or 180° could occur between two designs with similar dynamic mechanisms. This phase difference could excessively smooth the prediction results by traditional kriging. If the weighting numbers evaluated by kriging for these two cases are close and their instability waves are exactly out of phase, the wave information cancels out. As such, KSPOD applies a weighting function for the POD modes to ensure that similar POD modes in different cases can retain appropriate phase information. Nonetheless, the POD modes and coefficients are paired. The weighting number may fix the phase differences between spatial information, but it cannot adjust the phase differences between time information. Hence, some prediction results in KSPOD still exhibit time delays of evolving vortex structures.

Two cases per cluster, randomly selected to present the accumulated energy percentage of their pressure in the POD modes, are presented in Figure 47, with the accumulated energy percentage of pressure in the CKSPOD modes. As explained in Algorithm 3, all 30 training cases now share the same eigenvalues to affirm that no phase difference of coherent structures exists between any two designs. All 30 cases are now operated by the same eigen-decomposition process via the CKSPOD transfer matrix in Equation (34). Hence, the CKSPOD modes and energy accumulation will not perform similarly to the corresponding modes from the ordinary POD.

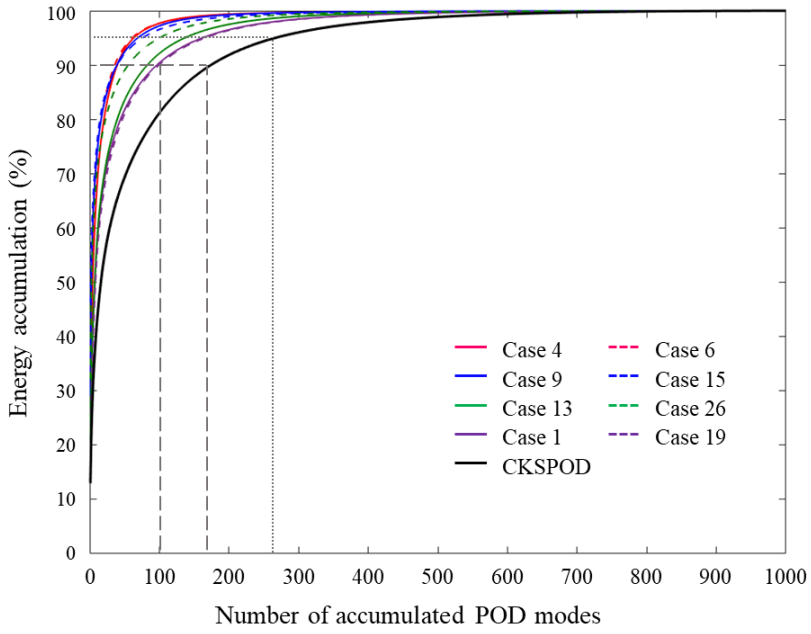

Figure 47—Energy accumulation of pressure POD modes

With ordinary POD, the first 23-99 modes contain over 90% of the total energy; the first 45-165 modes have over 95% of the energy; and the first 129-312 modes have over 99.9% of the energy. However, the CKSPOD transformation matrix with Hadamard superposes all flowfield information into one covariance matrix, which is latterly decomposed by eigen-decomposition and decreases the speed of energy accumulation. For CKSPOD, it reaches over 90% of the total energy after first 173 modes; the first 266 modes contain over 95% of the total energy; and the energy accumulation of pressure exceeds 99% after the first 522 modes. Moreover, the first CKSPOD mode only carries 12.99% of total energy; but the energy of all modes from ordinary POD ranges between 15.77% (Case 19 from Cluster D) and 52.76% (Case 24 from Cluster A) of total energy.

The slow speed of energy accumulation of CKSPOD implies that this algorithm requires more information than does KSPOD to collect sufficient energy to build a posterior model for the spatiotemporal flowfield prediction. As such, it is no longer an

assumption that the reduced data sets (that is, the dominant modes) capture the significant flow structures and their dynamic characteristics for the same rank with the same phase. The CKSPOD transform matrix in Equation (34) equips this criterion.

Cluster A contains more energy for the first and the most dominant POD mode, and CKSPOD decreases the speed of energy accumulation. Figure 48 compares the first four pressure modes between POD and CKSPOD from Case 16 in Cluster A. Here, all pressure modes are normalized and presented between -1 and 1. Mode 1 is an ensemble of the longitudinal modes of the hydrodynamic instability. In this case, the injector length is fixed at 25 mm, and the longitudinal hydrodynamic wave speed in the liquid film is estimated to be 7-9 m/s [10-12, 92]. This leads to a characteristic frequency of 0.32-0.45 kHz for the hydrodynamic instability. Mode 1 has similar structures between CKSPOD and POD, except that the energy carried by POD is approximately 2.5 times higher than CKSPOD. Mode 2 contains about 8.01% and 6.95% of the total energy from POD and CKSPOD, respectively. Although the two contour maps are not very similar, no significant energy difference exists between CKSPOD and POD in Mode 2. The energies obtained by POD in Modes 3 and 4 are around 1% greater than CKSPOD, and the distributions of these two modes from the two methods are alike. In short, Modes 1-3 capture similar physical characteristics in the dominant POD modes with the same order resulting from the eigen-decomposition [3]. The similarity of POD modes among cases in the same cluster justifies the two fundamental assumptions of KSPOD stated in Chang et. al. (2018) [3]. A transfer matrix is proposed in CKSPOD, as stated in Equation (34), to guarantee all POD modes with similarity will be decomposed with the same phase.

Figures 47 and 48 provide the evidence of energy loss by the CKSPOD transfer matrix and the similarity between POD and CKSPOD modes. To further check the similarity of base function information between CKSPOD and POD, Figure 49 compares the first four dominant modes of pressure for Case 16 in forms of probability density distribution. This distribution is obtained from the estimated kernel smoothing function. A kernel distribution is a nonparametric representation of the probability density function, $f_h(x)$, of a random variable, written as

$$f_h(x) = \frac{1}{nh}\sum_{i=1}^{n} \mathcal{K}\left(\frac{x-x_i}{bw}\right), \tag{68}$$

where n is the sample size, $\mathcal{K}(\cdot)$ is the density smoothing function, and bw is a smoothing parameter, bandwidth.

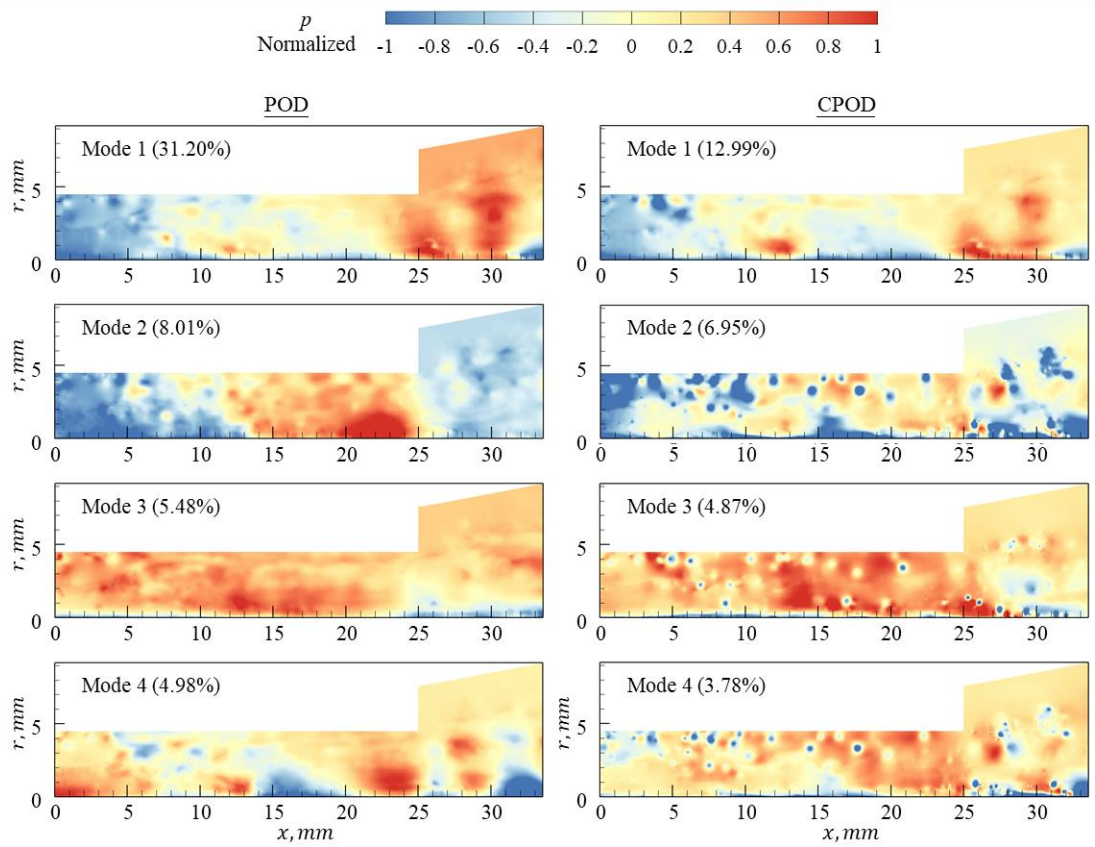

Figure 48—The first four dominant POD and CKSPOD modes of pressure for Case 16 (Cluster A)

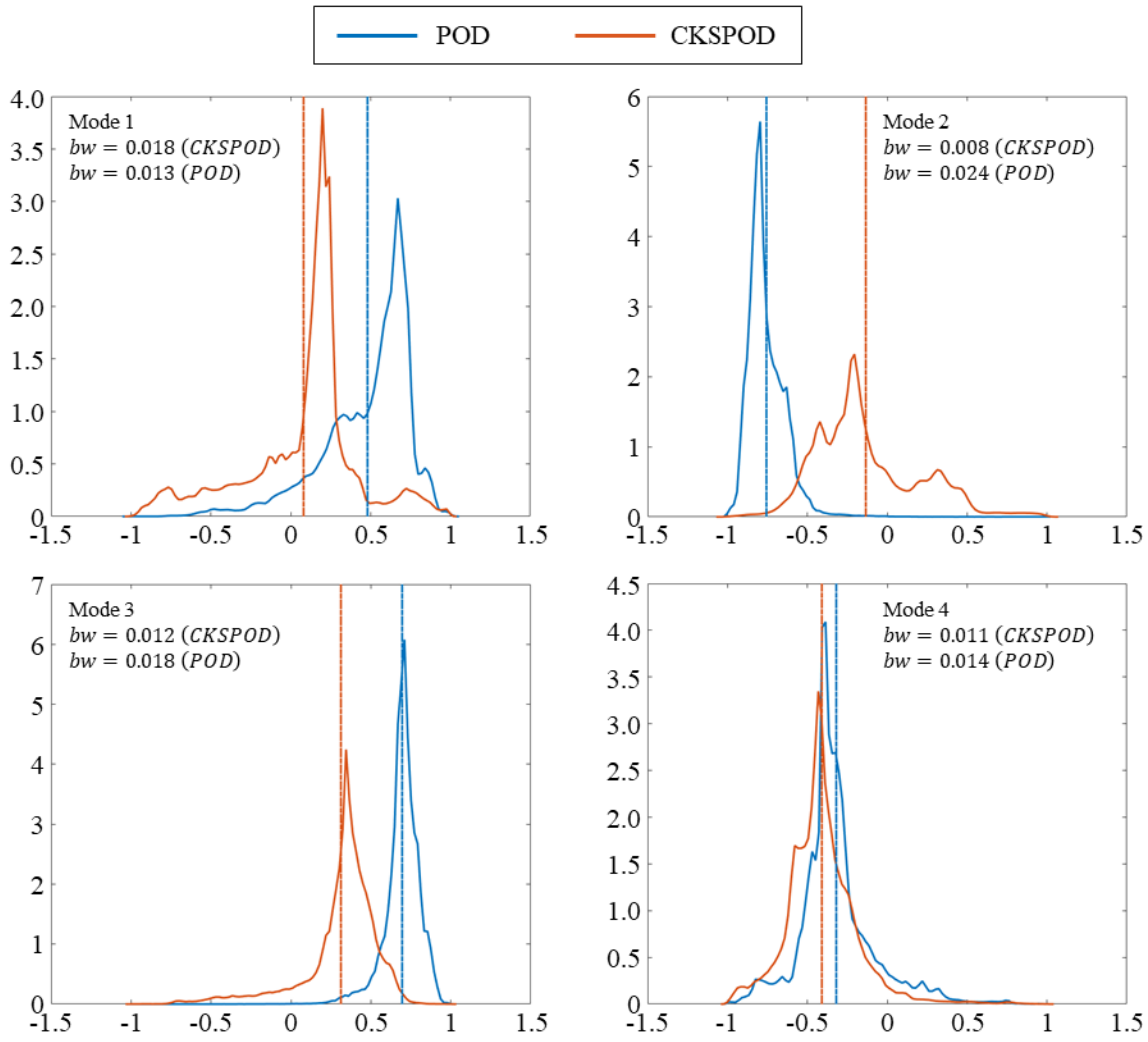

Figure 49 – Probability densities of POD and CKSPOD modes for Case 16 from Cluster A (vertical lines represent mean values and *bw* represents bandwidth of kernel smoothing function)

The results demonstrate that the probability density distribution curves of Mode 1 between CKSPOD and CPOD is similar, and no distinct difference exists between their bandwidths. From the normalized pressure in Figure 49 (i.e., the x-axial), approximately a 25% bias of mean values exists between POD and CKSPOD. Similar characteristics can be found in Modes 3 and 4, except that the normalized mean values of Mode 4 from CKSPOD and POD are very close. Mode 2 displays the most visible differences between

CKSPOD and POD. The bandwidth of probability density distribution for POD is over three times of the bandwidth from CKSPOD.

In summary, the mode contour shapes and probability density curves provide clear evidence that all physics information from training cases are transferred and relocated with a new set of eigenvalues by a CKSPOD transformation matrix. Nevertheless, all information is still well-preserved for the construction of emulation model.

4.4.2 Prediction by Common Kernel-Smoothed Proper Orthogonal Decomposition (CKSPOD) Based Emulation

Figure 50 shows an excellent comparison between the simulation result and prediction from the trained emulator for Test Case D2. The design parameters of test cases are given in Table 14. The CKSPOD-based emulator works well; it emulates essential flow structures successfully. The evolution of the liquid film and its spreading downstream of the injector exit also agree extremely well between the simulation and emulation. Although CKSPOD executes eigen-decomposition differently than KSPOD, the weighting number estimation process is the same for these two algorithms. The overall turnaround time for CKSPOD emulator is about 53 seconds of CPU time per snapshot, roughly 1.2 times longer than emulation by KSPOD. The CKSPOD requires more time for emulation because it requires more space and time information than KSPOD to obtain sufficient energy after data reduction to build the surrogate model.

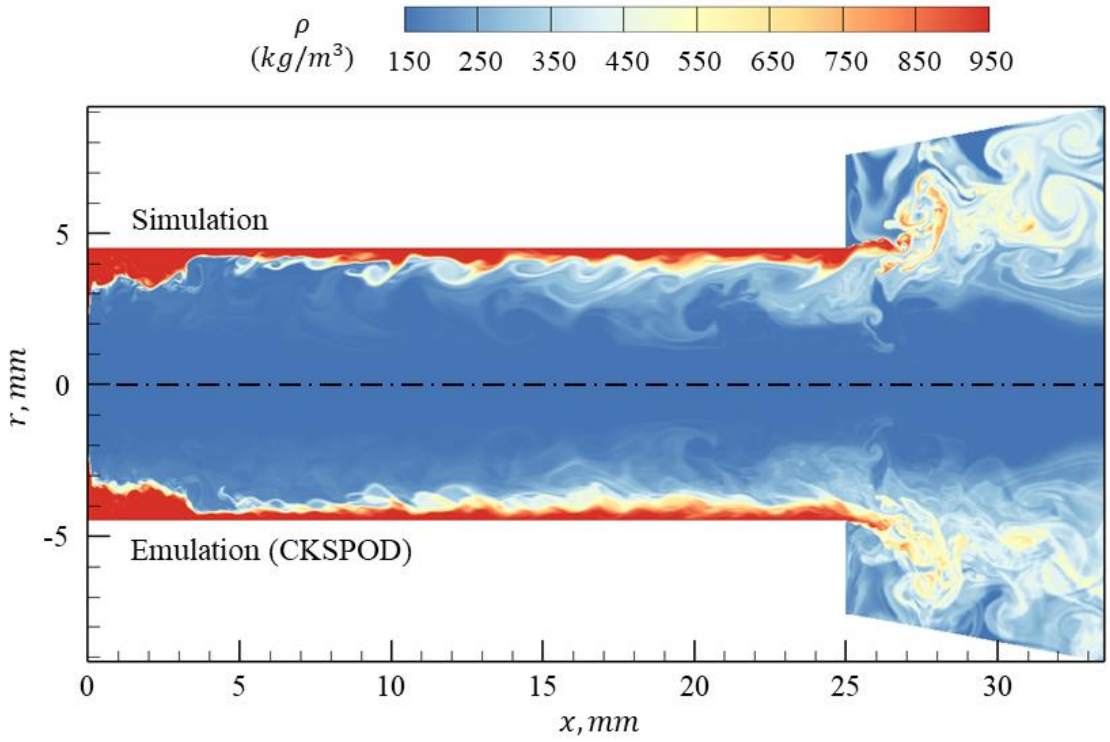

Figure 50—Comparison of density fields between LES-based simulation and predictions by CKSPOD-based emulation: Test Case D2 at *t* = 7.89 ms

Figures 51 and 52 illustrate the prediction results of CPOD, KSPOD, and CKSPOD for Test Cases A2 and C2. The intake velocity, u_{in}, for C2 is 22.33 m/s, which is over three times the value of Case A2, 6.65 m/s. These two figures show that even when there are over 400,000 mesh grids in the spatial domain for simulation, CPOD fails to capture detailed instantaneous flow dynamics. Conversely, KSPOD can emulate evolving flow dynamics well; however, the emulated vortex structure spreading downstream of the injector exit is not very similar to the corresponding simulation. Moreover, the detailed flow structures in the $r > 4.5$ mm downstream area become mist-like structures by KSPOD-based emulation, shown in Figure 51. However, by CKSPOD, the mist-like structures disappear and the emulated vortex structure spreading downstream of the injector exit much more resembles the corresponding simulation than the other two

methods. Figures 53 and 54 show more evidence of improvement of CKSPOD, using black and red dotted circles to demonstrate the most significant differences between CKSPOD-based and KSPOD-based emulation.

Figures 55 and 56 demonstrate the time evolution of flow dynamics of KSPOD-based and CKSPOD-based emulations for Test Case A2. Figure 55 shows the discrepancy of time delay caused by phase difference in KSPOD algorithm. In Figure 5 * MEREFORMAT 5, the traveling surface wave in the injector is shown to propagate downstream slightly faster in the emulation than in the simulation. The vortex structure predicted by KSPOD spreads downstream of the injector exit faster than the simulation as well. However, in the CKSPOD-based emulation shown in Figure 56, the traveling surface wave in the injector propagates almost synchronically with the wave in the simulation: toward downstream, to the injector exit. Figure 57 shows prediction results for all four clusters, in each of which two test cases are considered. The flow structures and dynamics are well captured by this new surrogate model, especially the mist-like turbulent structures that are barely seen in emulation results.

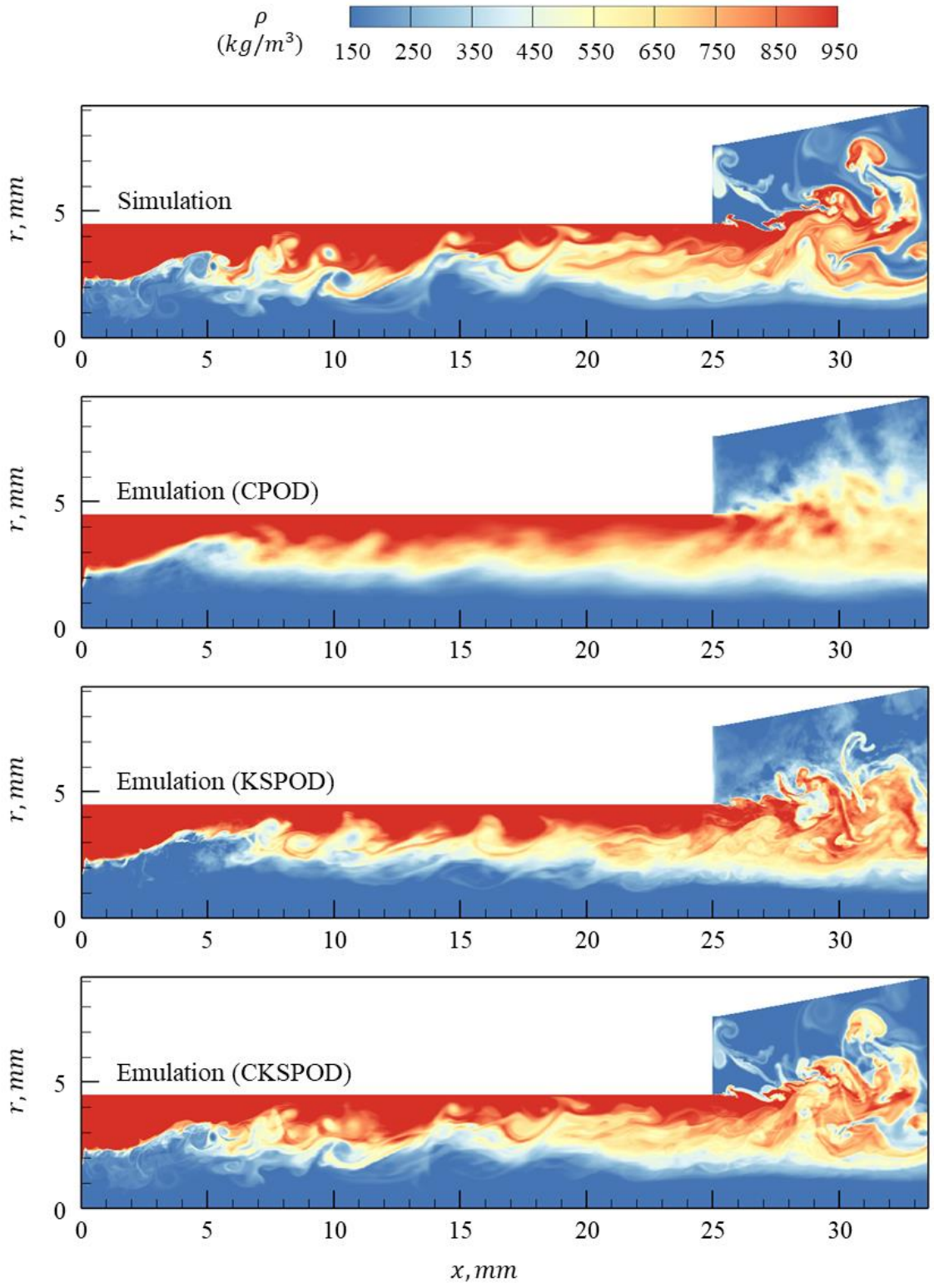

Figure 51—Comparison of density fields among LES-based simulation and predictions by three different emulations: Case A2 at *t* = 1.01 ms

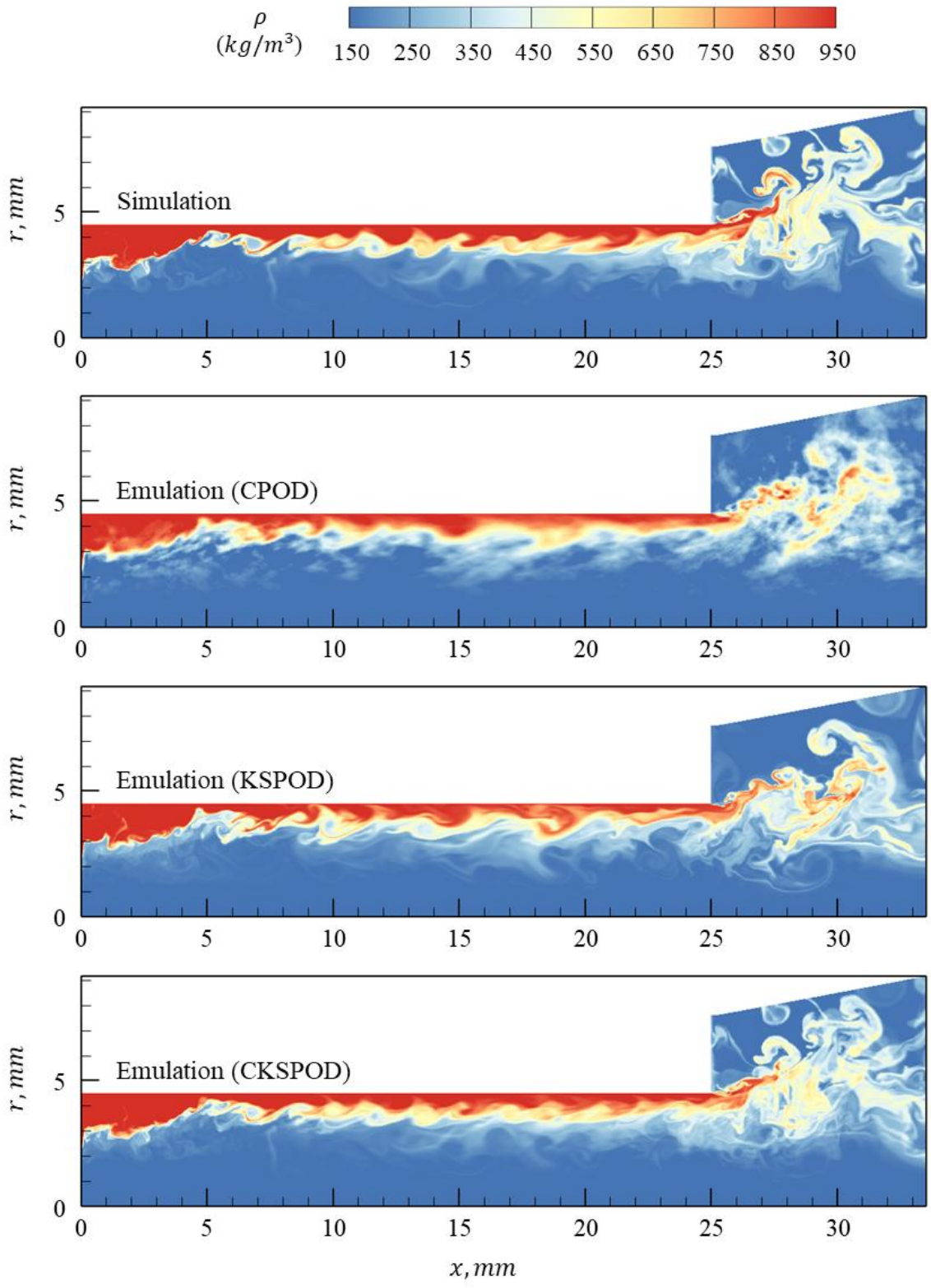

Figure 52—Comparison of density fields among LES-based simulation and predictions by three different emulations: Case C2 at *t* = 0.11 ms

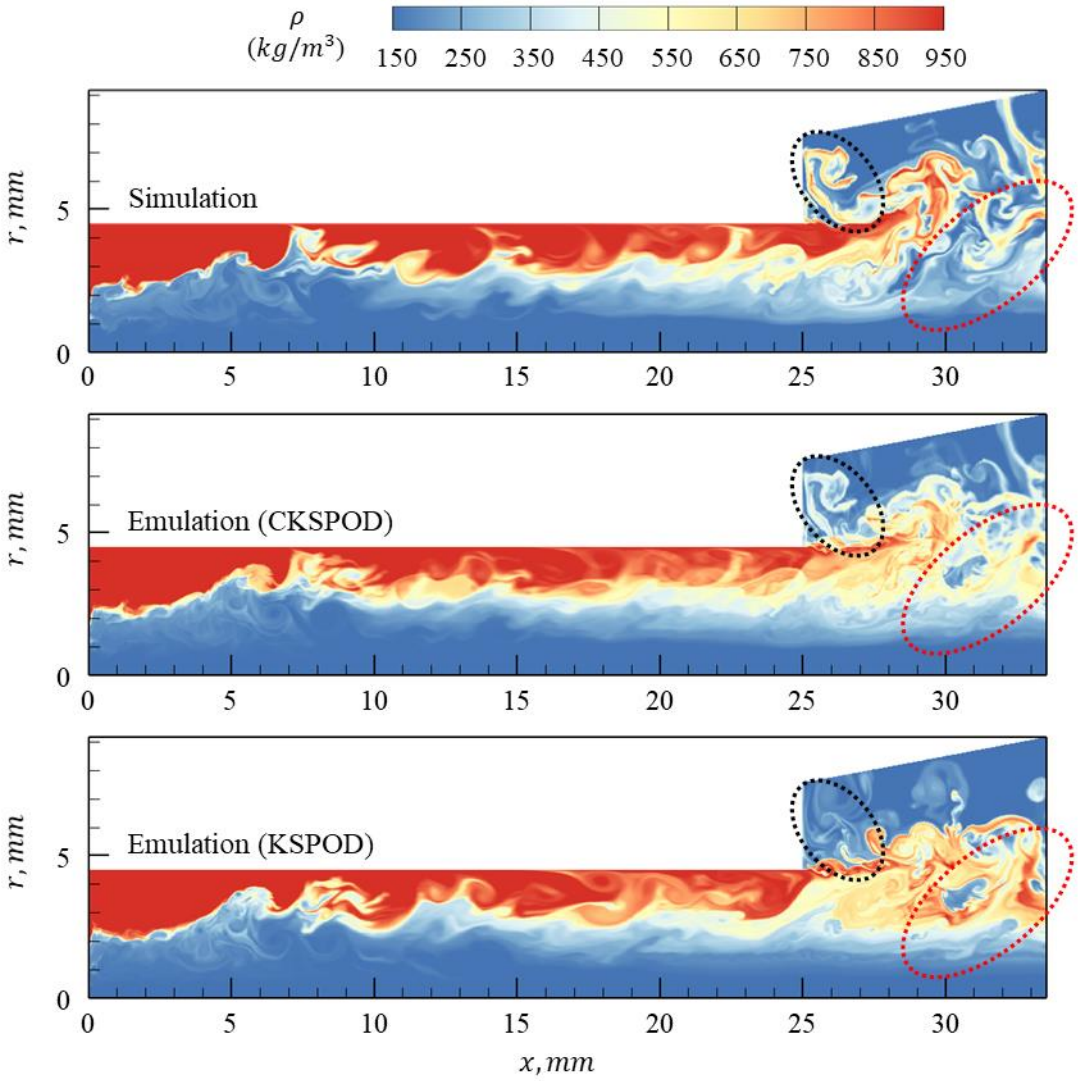

Figure 53—Comparison of density fields between LES-based simulation and prediction by KSPOD-based and CKSPOD-based emulations: Case B1 at *t* = 4.62 ms

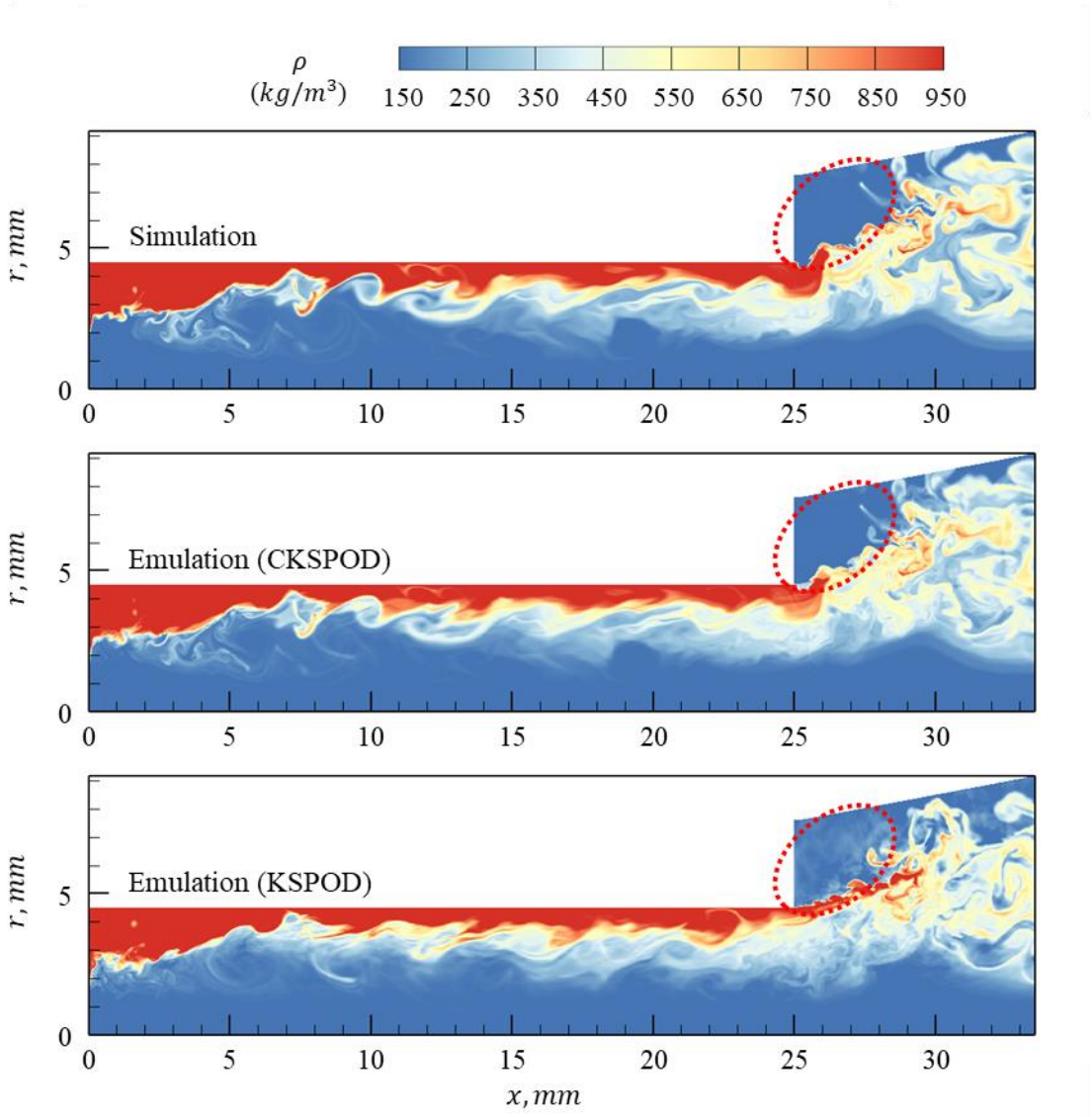

Figure 54—Comparison of density fields between LES-based simulation and prediction by KSPOD-based and CKSPOD-based emulations: Case B2 at *t* = 2.13 ms

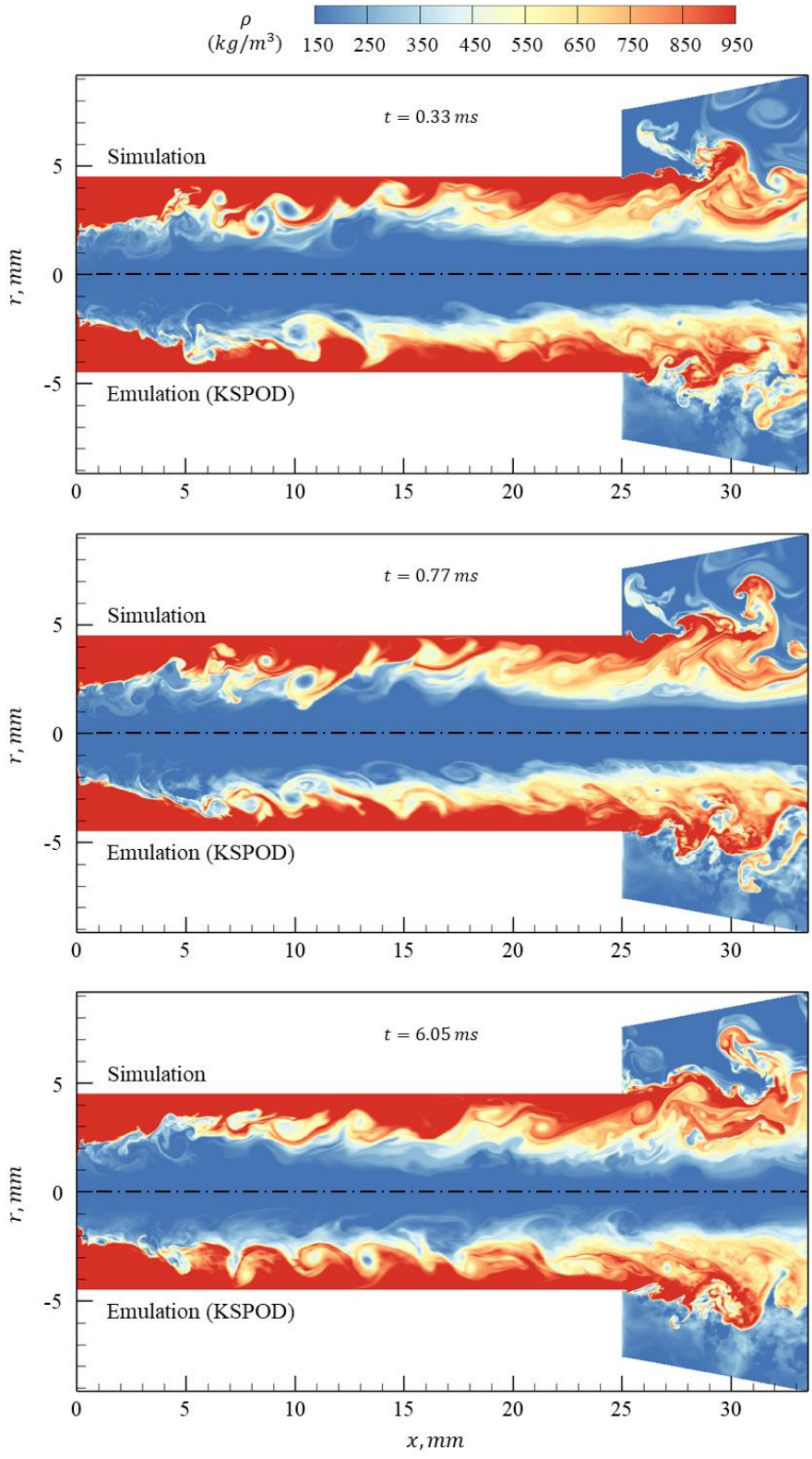

Figure 55—Comparison of density fields between LES-based simulation and KSPOD-based emulation: Case A2 at three different times

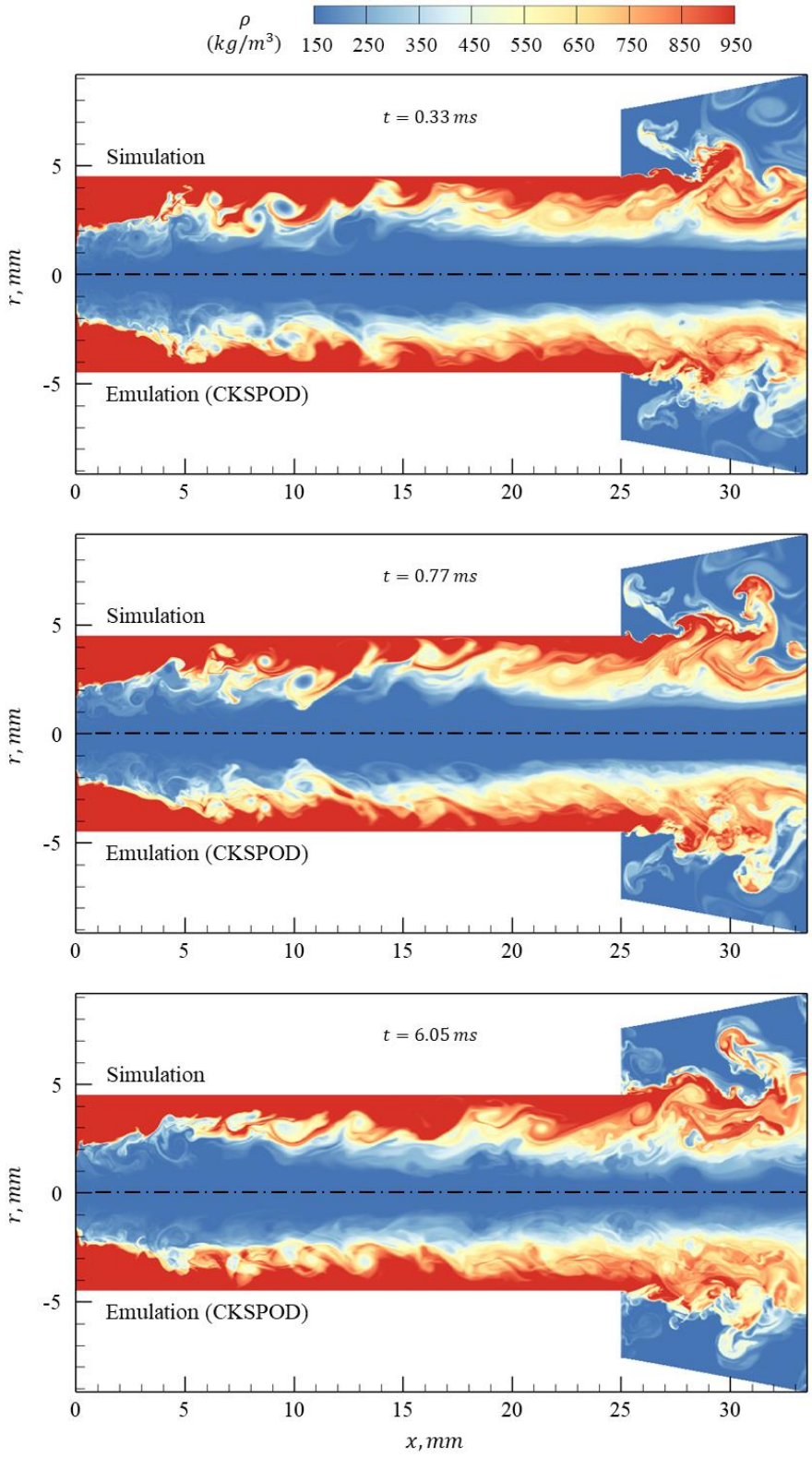

Figure 56—Comparison of density fields between LES-based simulation and CKSPOD-based emulation: Case A2 at three different times

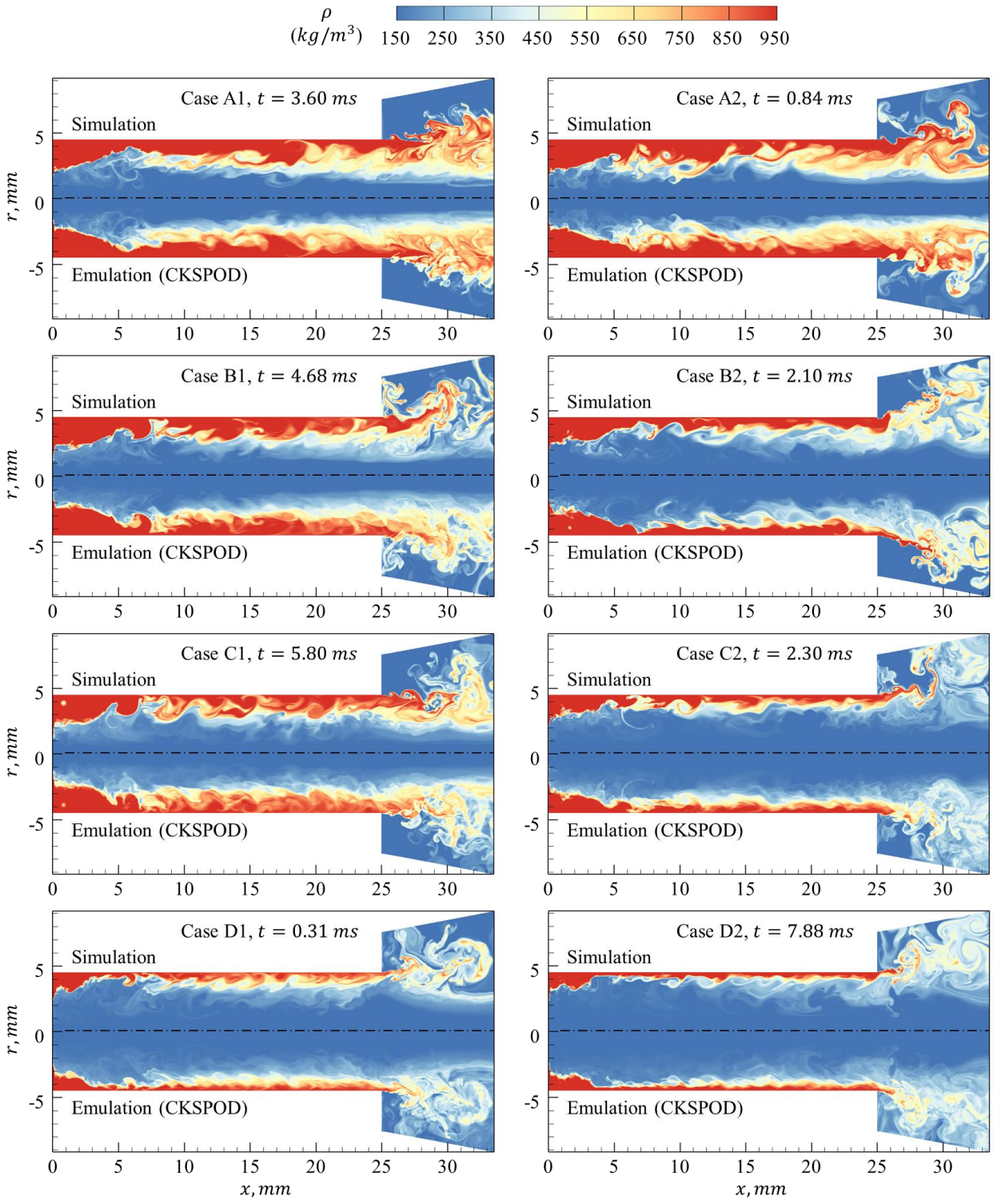

Figure 57—Comparison of density fields between LES-based simulation and CKSPOD-based emulation

To learn more about CKSPOD, an old set of 30 training cases for CPOD [1, 2] are also applied. As discussed in Chapter 2, CPOD studies with five design parameters, including injector length and radius, are shown in Figure 5 and Table 1. Since the design space of length and radius is wide, the 30 previous simulation runs can be divided into two subgroups, swirling (spreading angle above 30°) and jet-like (spreading angle below 30°) flows. Thus, one validation case is prepared for each group. Design space and validation case for all geometric parameters are tabulated in Table 17.

Table 17—Injector geometric parameters for design space and validation cases

	L (mm)	R_n (mm)	θ (°)	δ (mm)	ΔL (mm)
Training cases Design Space	20-100	2.0-5.0	45-75	0.5-2.0	1.0-4.0
Validation Case 1 (Swirl-like)	22.0	3.22	58.2	0.576	3.42
Validation Case 2 (Jet-like)	37.7	3.06	59.0	1.417	1.00

Figures 58 and 59 present the instantaneous density distribution of Validation Cases 1 and 2, respectively, emulated by CKSPOD. All black-dotted lines label the dissection method of the original common-grid idea, presented by Shiang et. al. [2]. In our previous work, the CPOD-based surrogate model could only emulate time-mean property distribution features. It must be noted that the 30 previous simulation runs were obtained by a grid of 100,000 mesh points; the 30 new simulation cases, designed and adopted for KSPOD and CKSPOD, were calculated by a grid of over 400,000 mesh points. Also, the total numbers of simulation runs were decided by a 6*d* rule, not 10*d*, based on the cost of time. Nevertheless, Figure 51 and Figure 52 prove that CPOD fails to capture instantaneous flow information after increasing the density of simulation mesh points and decreasing the design space and number of parameters.

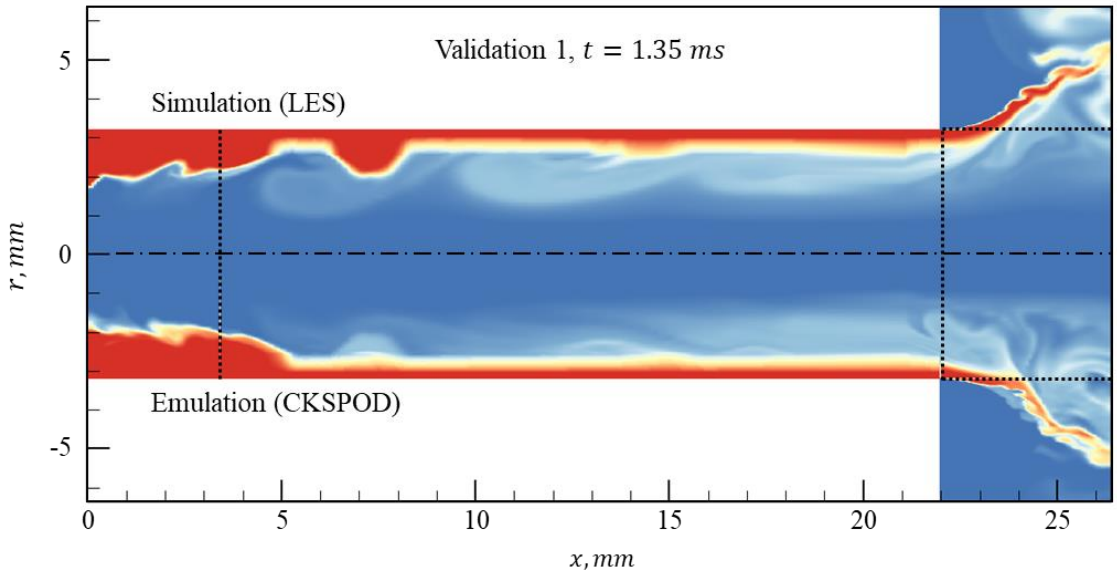

Figure 58—Comparison of density fields between LES-based simulation and CKSPOD-based emulation: Validation Case 1 (swirl-like flow) from 30 previous cases built for CPOD (dotted lines label the dissection method of the common grid)

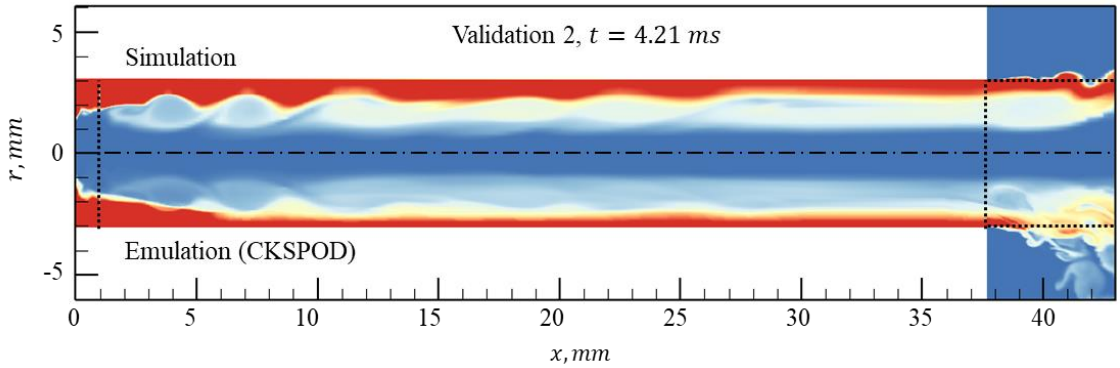

Figure 59—Comparison of density fields between LES-based simulation and CKSPOD-based emulation: Validation Case 2 (jet-like flow) from 30 previous cases built for CPOD (dotted lines label the dissection method of the common grid)

In CKSPOD, the emulation is governed with large design space better than in CPOD. The CKSPOD-based emulation performs better in the swirl-like case (Figure 58) than in the jet-like case (Figure 59). Although CKSPOD does not capture the flow structures spreading downstream of the injector exit in jet-like case very well, it still

demonstrates a possible capability as a surrogate model to predict spatiotemporal evolving flow dynamics with wide design space. The CKSPOD-based emulation can be improved for large design space cases given sufficient numbers of training cases.

4.4.3 Accuracy of Prediction: Film Thickness and Spreading Angle

To further evaluate the accuracy of the CKSPOD-based emulation, two performance measures, film thickness and spreading angle, are assessed. Table **18** presents a comparison of the time-mean simulation and two different emulation (prediction) results, obtained by averaging the instantaneous data over a statistically meaningful time duration. The error is calculated as follows:

$$\varepsilon_{abs} = \frac{|x_{sim} - x_{emu}|}{x_{sim}} \times 100\%, \tag{69}$$

where x_{sim} represents data from simulation and x_{emu} from emulation. Detailed information about mean values, standard deviations, and averaged absolute error $\overline{\varepsilon_{abs}}$ is given in Table 18. The results reveal that CKSPOD achieves a comprehensive improvement of the accuracy of liquid-film thickness and spreading. Most errors of CKSPOD emulation are under 1%. The maximum absolute errors for liquid-film thickness and spreading angle are 0.36 % in Case D1 and 0.41% in Case A2, respectively, which displays the same trends as in KSPOD; the only exception is that CKSPOD has a larger error than KSPOD in the spreading angle of liquid film in Test Case C1. As most training cases are created in Clusters A and B, the predictions are surely more robust within these two clusters. Hence, missing training cases for Cluster C could explain why CKSPOD does

not perform better than KSPOD in predicting the spreading angle of liquid film, especially since the error of CKSPOD for Case C1 is still small.

Figures 60 and 61 compare the probability density distributions of spreading angle and liquid-film thickness within simulations and two emulations for all test cases. The vertical lines represent mean values. The probability densities of instantaneous spreading angle and liquid-film thickness for all validation cases are estimated by kernel smoothing function in Equation (68). These two figures show that the probability density curves of instantaneous spreading angle and liquid-film thickness in CKSPOD resemble those from simulation better than does KSPOD. Also, the widths of most CKSPOD probability density curves are smaller than those from KSPOD, which explains why CKSPOD has a smaller standard deviation of two measurement performance predictions than does KSPOD.

Another method for measuring the performance of the emulation examines the distribution of liquid-film thickness along the axial direction. Figure 62 illustrates a comparison within results averaged over 1,000 snapshots from the simulation, CKSPOD-based emulation, and KSPOD-based emulation. Figure 63 exhibits the absolute errors from the simulations and the two emulations. The horizontal lines in Figure 63 represent the averaged absolute error, $\overline{\varepsilon_{abs}}$, for each test case. The liquid-film thickness predicted by the CKSPOD-based emulation has an averaged error less than 3% for all cases, which performs much better than the KSPOD-based emulation, which displays $\overline{\varepsilon_{abs}} = 6.4\%$ for Case B2. For both KSPOD and CKSPOD, cases with higher inlet velocities have less variation for the liquid-film thickness near the injector exit. However, the first local maximum of error occurs in the LOX inlet area, which contains the highest momentum and kinetic energy, for KSPOD, but does not produce obvious results for CKSPOD. The second peak of error

occurs when the flowfield is still developing. In this region, both KSPOD and CKSPOD show significant errors along the injector.

The large error along the axial direction in Case B2 from KSPOD can be attributed to this case being close to the boundary between Clusters B and C and including more flow mechanisms than are prominent in the cases in Cluster C. Since CKSPOD fixes the phase differences of physics coherent structures among all training cases, this large error along the axial direction in Case B2 is successfully decreased from 6.4% to 2.93%.

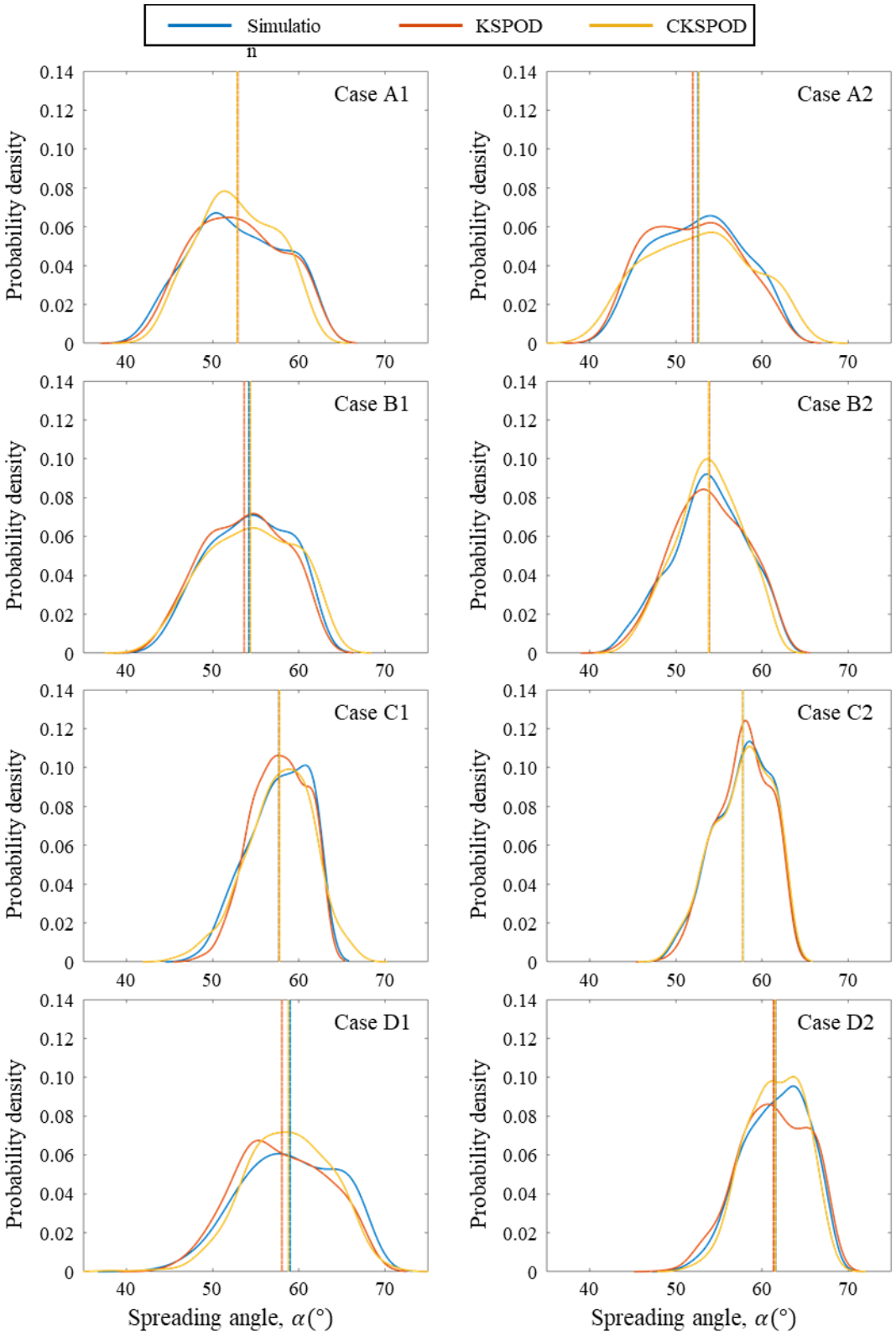

Figure 60—Probability densities of instantaneous spreading angle for test cases (vertical lines represent mean values)

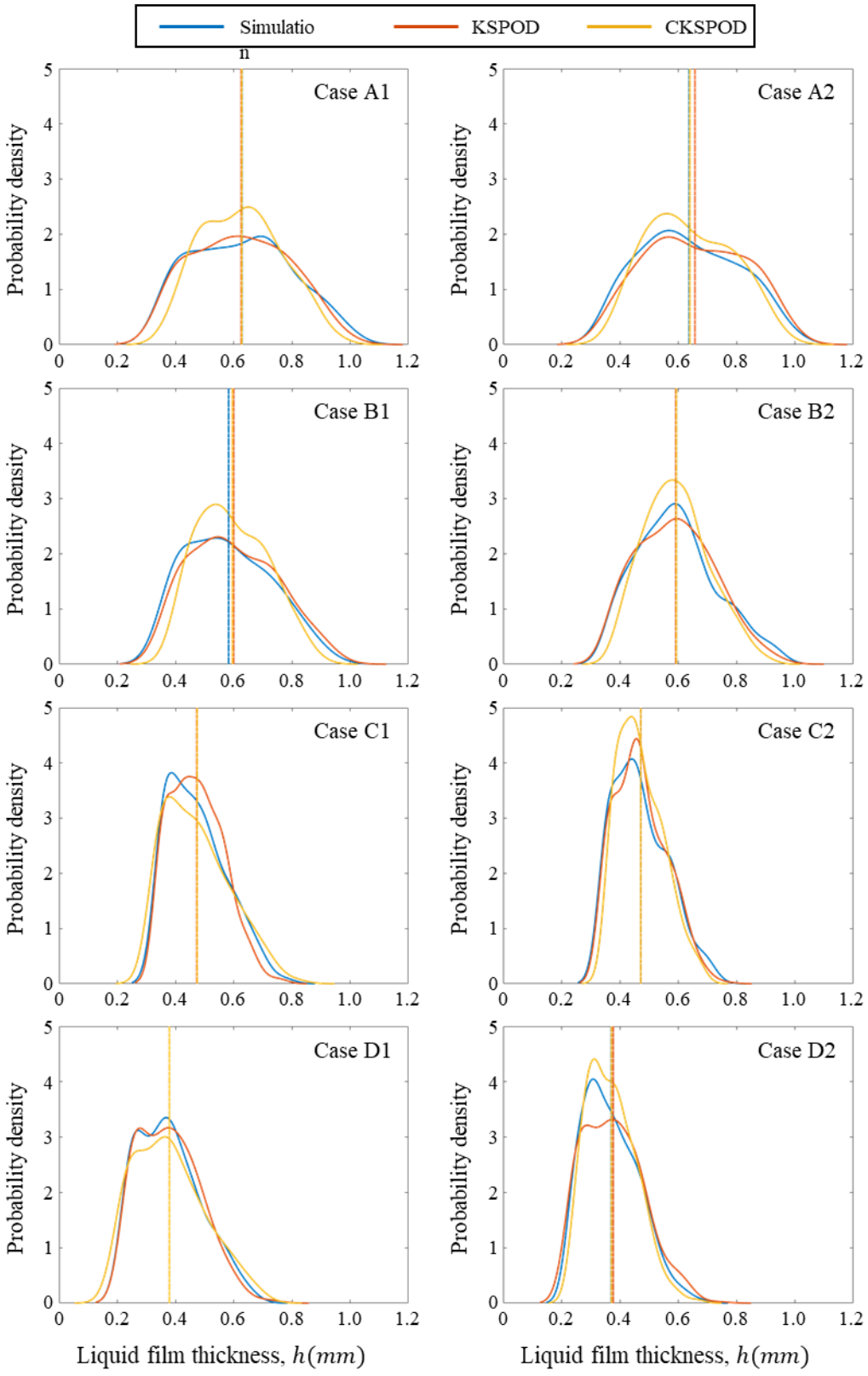

Figure 61—Probability densities of instantaneous liquid-film thickness for test cases (vertical lines represent mean values)

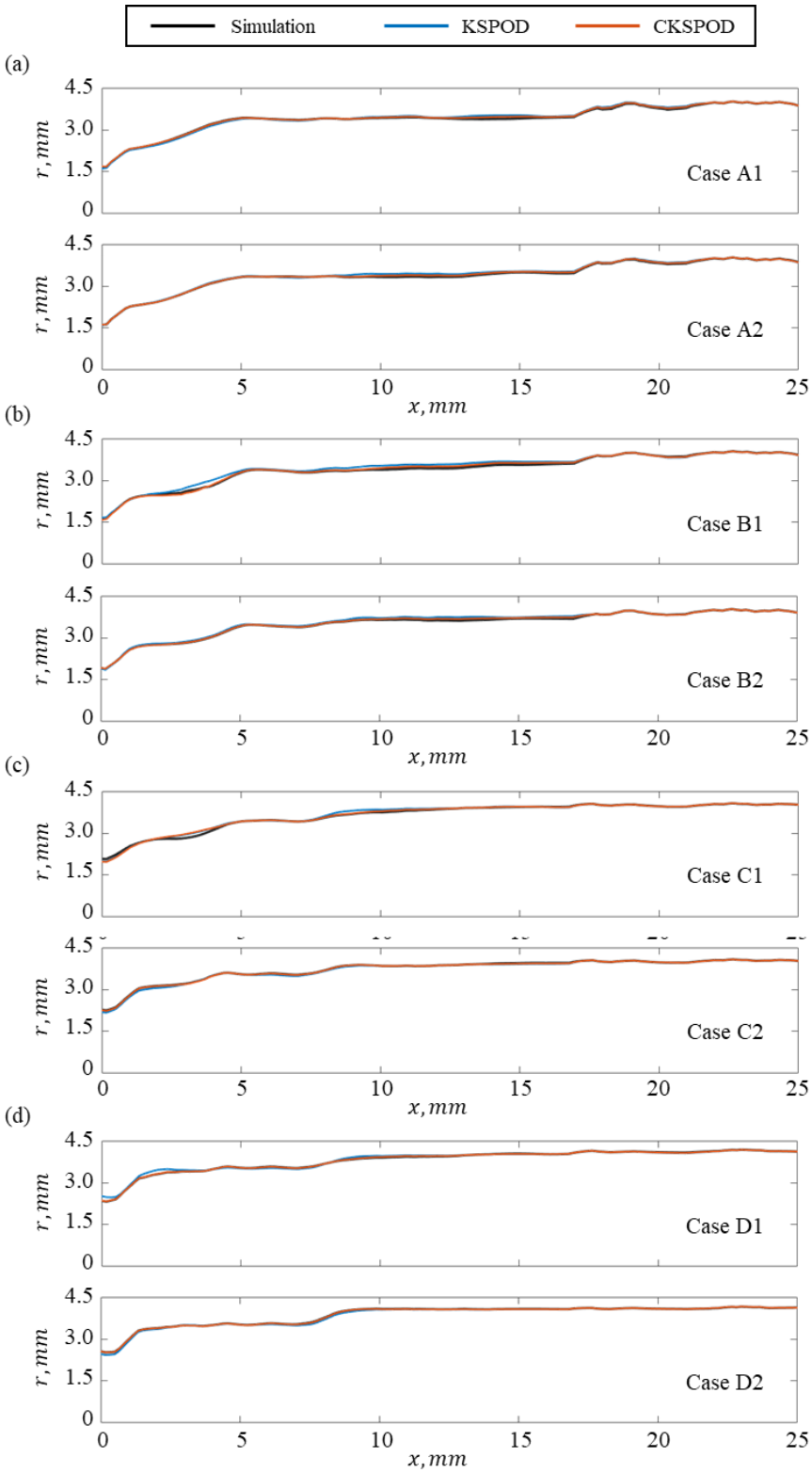

Figure 62—Comparison of liquid-film thickness along the axial direction, averaged over 1,000 snapshots

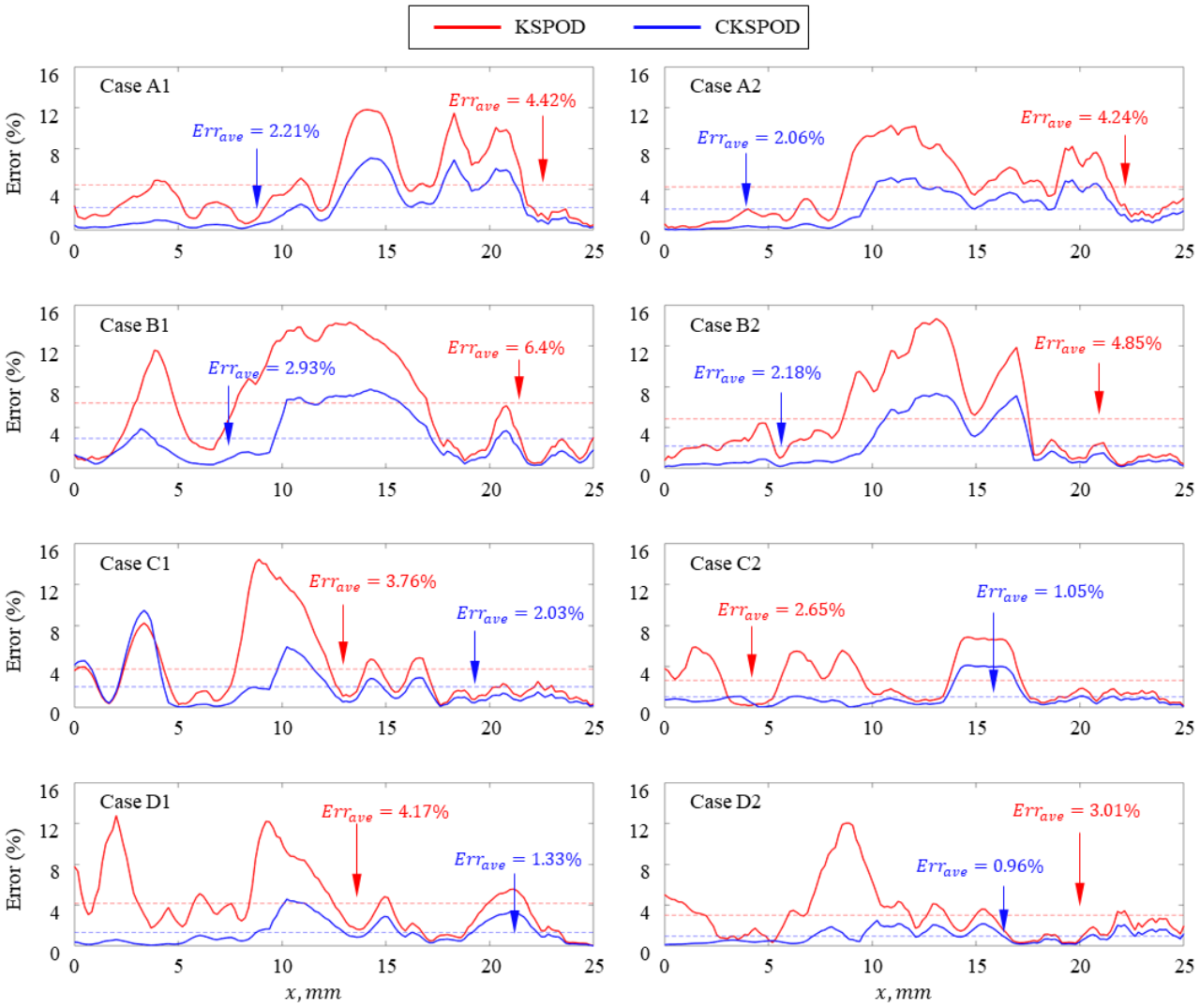

Figure 63—Error for liquid-film thickness along the axial direction

Table 18—Film thickness and spreading angle for simulation and emulation results

Case Number	Analysis	Spreading Angle (°)			Film Thickness (mm)		
		Simulation	KSPOD	CKSPOD	Simulation	KSPOD	CKSPOD
A1	Average	52.846	52.919	52.857	0.629	0.625	0.628
	STD	5.185	4.976	4.392	0.169	0.162	0.136
	Error	-	0.14%	0.02%	-	0.51%	0.10%
A2	Average	52.566	51.959	52.657	0.637	0.657	0.640
	STD	5.028	5.016	5.897	0.165	0.166	0.144
	Error	-	1.15%	0.17%	-	3.14%	0.41%
B1	Average	54.216	53.660	54.373	0.582	0.600	0.595
	STD	4.542	4.546	4.969	0.145	0.146	0.119
	Error	-	1.02%	0.29%	-	3.03%	2.25%
B2	Average	53.811	53.875	53.819	0.594	0.592	0.594
	STD	4.226	4.130	3.732	0.136	0.132	0.111
	Error	-	0.12%	0.02%	-	0.40%	0.04%
C1	Average	57.684	57.713	57.758	0.474	0.473	0.475
	STD	3.415	3.086	3.800	0.100	0.089	0.112
	Error	-	0.05%	0.13%	-	0.36%	0.04%
C2	Average	57.778	57.741	57.750	0.471	0.472	0.471
	STD	3.177	3.016	3.244	0.093	0.087	0.077
	Error	-	0.06%	0.05%	-	0.13%	0.02%
D1	Average	58.998	58.031	58.786	0.379	0.379	0.379
	STD	5.389	5.146	4.860	0.107	0.105	0.120
	Error	-	1.64%	0.36%	-	0.02%	0.10%
D2	Average	61.586	61.334	61.541	0.370	0.377	0.371
	STD	3.617	3.893	3.289	0.094	0.101	0.083
	Error	-	0.41%	0.07%	-	1.97%	0.26%

4.5 Uncertainty Quantification (UQ) Analysis

4.5.1 Temporal Uncertainty Quantification (UQ) Model

For computer experiments, the quantification of predictive uncertainty can be as crucial as the prediction itself. To this end, a spatiotemporal representation of this UQ is provided, which is shown to have a useful and appealing physical interpretation. For spatial UQ, the top plot of Figure 64 shows the one-sided width of the 80% pointwise CI from Equation (51), for x-velocity at $t = 15$ ms. The emulator is demonstrated to be most certain in predictions near the inlet and centerline of the injector; conversely, high predictive uncertainty exists at the three gaseous cores downstream (shown in green in Figure 64). This is logical in the physical sense, because these cores correspond to flow recirculation vortices and therefore exhibit highly unstable flow behavior. From the bottom plot of Figure 64, which shows the absolute emulation error of the same flow, the pointwise confidence band not only covers the realized prediction error but also roughly mimics its spatial distributions.

For temporal UQ, Figure 65 shows the same one-sided CI width at probe 1 (see Figure 34). This temporal uncertainty is relatively steady over *t*, except for two abrupt spikes at timesteps around 300 and 800. These two spikes have an appealing physical interpretation: the first indicates a flow displacement effect of the central vortex core, whereas the second can be attributed to the boundary development of the same core. This again demonstrates the usefulness of UQ not only as a measure of predictive uncertainty but also as a means for extracting useful flow physics without the need for expensive simulations.

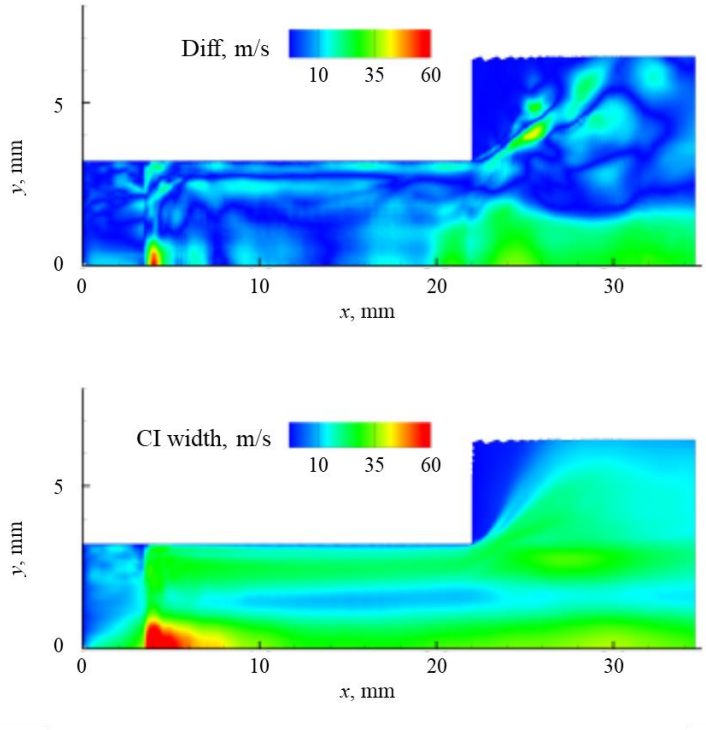

Figure 64—Absolute prediction error (top) and pointwise CI width (bottom) for x-velocity at t = 15 ms

To illustrate the improved UQ of the proposed model (see **Theorem 1**), the derived quantity TKE is used, which is typically defined as

$$\kappa(\boldsymbol{X},t)=\frac{1}{2}\sum_{r\in(u,v,w)}\left\{Y^{(r)}(\boldsymbol{X},t)-\bar{Y}^{(r)}(\boldsymbol{X})\right\}^2, \tag{70}$$

where $Y^{(u)}(\boldsymbol{X},t)$, $Y^{(v)}(\boldsymbol{X},t)$, and $Y^{(w)}(\boldsymbol{X},t)$ are flows for x, y, and circumferential velocities, with $\bar{Y}^{(u)}(\boldsymbol{X})$, $\bar{Y}^{(v)}(\boldsymbol{X})$, and $\bar{Y}^{(w)}(\boldsymbol{X})$ representing the corresponding time-averages. Such a quantity is particularly important for studying turbulent instabilities, because it measures fluid rotation energy within eddies and vortices.

For the sake of simplicity, it is assumed that (a) the time-averages $\bar{Y}^{(u)}(\boldsymbol{X})$, $\bar{Y}^{(v)}(\boldsymbol{X})$, and $\bar{Y}^{(w)}(\boldsymbol{X})$ are fixed and (b) the parameters $(\boldsymbol{\mu}, \mathbf{T}, \boldsymbol{\tau})$ are known. The following theorem provides the MMSE predictor and pointwise CI for $\kappa(\boldsymbol{X}, t)$ (proof in Appendix C):

Theorem 3. For fixed $\boldsymbol{X}$ and $\boldsymbol{t}$, the MMSE predictor of $\boldsymbol{\kappa}(\boldsymbol{X}, \boldsymbol{t})$ at a new setting $\boldsymbol{c_{new}}$ is

$$\hat{\kappa}(\boldsymbol{X}, t) = \frac{1}{2} \sum_{r \in (u,v,w)} \left\{ \hat{Y}^{(r)}(\boldsymbol{X}, t) - \bar{Y}^{(r)}(\boldsymbol{X}) \right\}^2 + tr\{\Phi(\boldsymbol{X}, t)\}, \tag{71}$$

where $\hat{Y}^{(r)}(\boldsymbol{X}, t)$, $\hat{Y}^{(r)}(\boldsymbol{X}, t)$, *and* $\hat{Y}^{(r)}(\boldsymbol{X}, t)$ *are the predicted flows for* x, y, *and circumferential velocities from (9), and* $\Phi(\boldsymbol{X}, t)$ *is defined in Equation* (75). *Moreover,* $\hat{\kappa}(\boldsymbol{X}, t)$ *is distributed as a weighted sum of non-central* χ^2 *random variables, with an explicit expression given in Equation* (78)*, in Appendix C.*

In practice, plug-in estimates are used for both time-averaged flows and model parameters.

Proof*:* For some spatial coordinate **x** and time step t, and let

$$\boldsymbol{y} = \left(Y^{(u)}(\mathbf{x}, t; \boldsymbol{c}_{new}), Y^{(v)}(\mathbf{x}, t; \boldsymbol{c}_{new}), Y^{(w)}(\mathbf{x}, t; \boldsymbol{c}_{new}) \right)^T \tag{72}$$

be the true simulated flows for *x*-, *y*-, and circumferential velocities at the new setting $\boldsymbol{c}_{new}$,

$$\hat{\boldsymbol{y}} = \left(\hat{Y}^{(u)}(\mathbf{x}, t; \boldsymbol{c}_{new}), \hat{Y}^{(v)}(\mathbf{x}, t; \boldsymbol{c}_{new}), \hat{Y}^{(w)}(\mathbf{x}, t; \boldsymbol{c}_{new}) \right)^T \tag{73}$$

be its corresponding prediction from Equation (48), and

$$\overline{\boldsymbol{y}} = \left(\overline{Y}^{(u)}(\mathbf{x}; \boldsymbol{c}_{new}), \overline{Y}^{(v)}(\mathbf{x}; \boldsymbol{c}_{new}), \overline{Y}^{(w)}(\mathbf{x}; \boldsymbol{c}_{new})\right)^T \tag{74}$$

be its time-averaged flow. It can be simply verified that, given the simulation data $\mathcal{D} = \left\{Y^{(r)}(\mathbf{x}, t; \boldsymbol{c}_i)\right\}$, the conditional distribution of $\boldsymbol{y}|\mathcal{D}$ is $\mathcal{N}(\widehat{\boldsymbol{y}}, \Phi(\mathbf{x}, t))$, where

$$\boldsymbol{\Phi}(\mathbf{x}, \boldsymbol{t}) \equiv \begin{bmatrix} \mathbf{m}^{(u)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{m}^{(v)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{m}^{(w)} \end{bmatrix} [\mathbb{V}\{\boldsymbol{\beta}(\boldsymbol{c}_{new})|\{\boldsymbol{\beta}(\boldsymbol{c}_i)\}_{i=1}^{n}\}]_{uvw} \begin{bmatrix} \mathbf{m}^{(u)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{m}^{(v)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{m}^{(w)} \end{bmatrix}^T, \tag{75}$$

with

$$\mathbf{m}^{(u)} = \left[\boldsymbol{\mathcal{M}}_{new}\left\{\boldsymbol{\phi}_1^{(r)}(\mathbf{x})\right\}, \boldsymbol{\mathcal{M}}_{new}\left\{\boldsymbol{\phi}_2^{(r)}(\mathbf{x})\right\}, \ldots, \boldsymbol{\mathcal{M}}_{new}\{\boldsymbol{\phi}_{K_r}^{(r)}(\mathbf{x})\}\right], \quad \boldsymbol{r} = \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}. \tag{76}$$

If $\Phi(t) = \mathbf{U}\Lambda\mathbf{U}^T$ is the eigen-decomposition of $\Phi(t)$, with $\Lambda = \mathrm{diag}\{\lambda_j\}$, it follows that $\Lambda^{-1/2}\mathbf{U}^T(\boldsymbol{y} - \overline{\boldsymbol{y}})|\mathcal{D} \overset{\text{def}}{=} \mathcal{N}(\boldsymbol{\mu}, \mathbf{I}_K)$, where $\boldsymbol{\mu} = \Lambda^{-1/2}\mathbf{U}^T(\widehat{\boldsymbol{y}} - \overline{\boldsymbol{y}})$ and $K = K_u + K_v + K_w$. Denoting $\mathbf{a} = \Lambda^{-1/2}\mathbf{U}^T(\boldsymbol{y} - \overline{\boldsymbol{y}})$, the TKE expression in Equation (71) can be written as

$$\kappa(\mathbf{x}, t) = \frac{1}{2}(\boldsymbol{y} - \overline{\boldsymbol{y}})^T(\boldsymbol{y} - \overline{\boldsymbol{y}}) = \frac{1}{2}\left(\mathbf{U}\Lambda^{1/2}\mathbf{a}\right)^T(\mathbf{U}\Lambda^{1/2}\mathbf{a})$$

$$= \frac{1}{2}\left(\mathbf{a}^T \Lambda^{1/2} \mathbf{U}^{\boldsymbol{T}} \mathbf{U} \Lambda^{1/2} \mathbf{a}\right)$$

$$= \frac{1}{2}\mathbf{a}^T \Lambda \mathbf{a} = \frac{1}{2}\sum_{j=1}^{K} \lambda_j a_j^2. \tag{77}$$

Since $\mathbf{a} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{I}_K)$, a_j^2 has a non-central chi-square distribution with one degree of freedom and non-centrality parameter μ_j^2 (denoted as $\chi_1^2(\mu_j^2)$), $\kappa(\mathbf{x}, t)$ then becomes

$$\sum_{j=1}^{K} \frac{\lambda_j}{2} \chi_1^2\left(\mu_j^2\right), \tag{78}$$

which is a sum of weighted non-central chi-squared distributions. This computation of the distribution function for such a random variable has been studied extensively [93-97], and these methods are utilized for computing the pointwise CI of $\kappa(\mathbf{x}, t)$ in Section 4. Specifically, the method of Liu et al. (2009) [97] is employed through the R [98] package CompQuadForm [99]. □

With this in hand, the prediction and UQ of TKE from the proposed model M_A and the independent model M_0 (see Theorem 1) are compared with the simulated TKE at the validation setting. Figure 66 shows the predicted TKE $\hat{\kappa}(\boldsymbol{X}, t)$ at Probe 8 (see Figure 34) over the fully developed time-frame of $t = 15 - 30$ ms, along with the 90% lower pointwise confidence band constructed using Theorem 3. Visually, the proposed model M_A provides an improved prediction fo the simulated TKE than does the independent model M_0. Regarding the confidence bands, the average coverage rate for M_A over the fully developed time-frame (85.0 %) is much closer to the desired nominal rate of 90%, compared to that for M_0 (73.8%). The proposed model therefore provides a coverage rate

closer to the desired nominal rate of 90%. The poor coverage rate for the independent model is shown in the right plot of Figure 66, where the simulated TKE often dips below the lower confidence band. By incorporating prior knowledge of flow couplings, the proposed model can provide improved predictive performance and UQ.

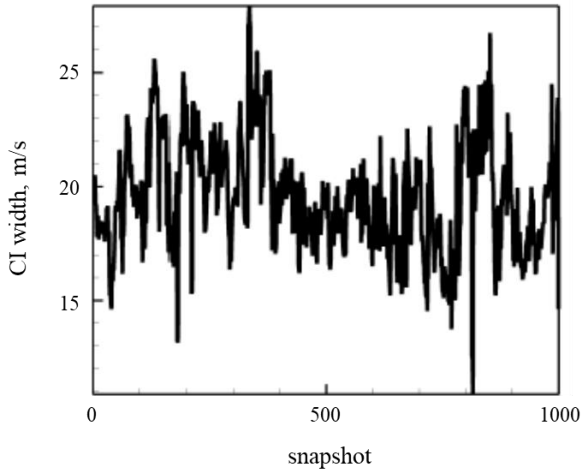

Figure 65—CI width of x-velocity at Probe 1

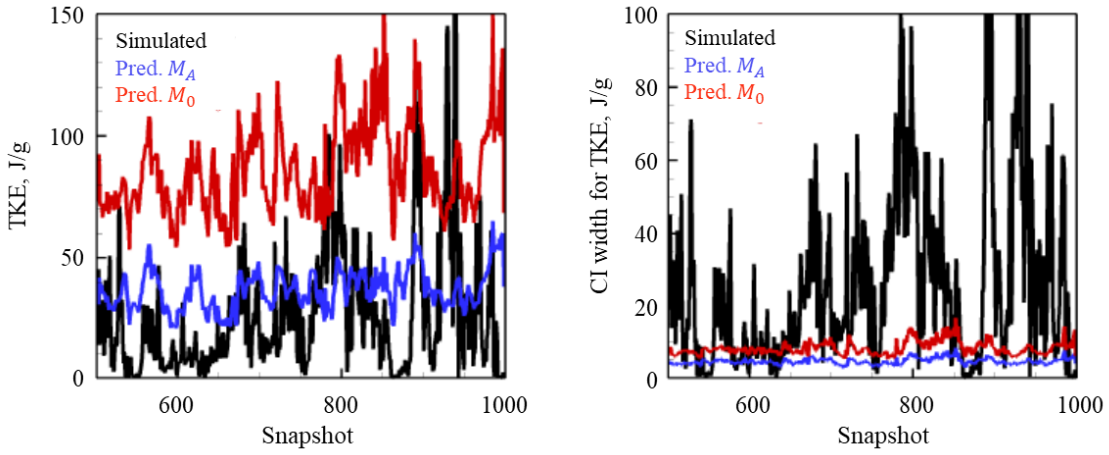

Figure 66—Predicted TKE and lower 90% confidence band for models M_A and M_0 at Probe 8

4.5.2 Uncertainty Quantification (UQ) Comparison Between Two Emulators

Since Section 3.3 provides much evidence that CKSPOD and KSPOD have better accuracy than CPOD on predictions of visualized flowfields and performances of measurement, it is important to learn the UQ of the CKSPOD and KSPOD models. In the previous section, the proposed model, M_A, provides an improved prediction flow the simulated TKE than the does the independent model, M_0. Therefore, M_A is applied to calculate the 90% CI for both KSPOD and CKSPOD, and Theorem 3 is applied to calculate the predicted TKE. The variance of time coefficients is derived from Equations (60) and(63).

Figures 67 to 74 demonstrate the contour map of temporal averaged TKE and standard deviation for CKSPOD and KSPOD for all eight validation cases. The standard deviation is calculated from the variance based on Equations (49) and(51). In these figures, the temporal averaged TKE of CKSPOD is closer to the temporal averaged TKE of simulation than is KSPOD. The smaller the variance, the greater the precision. In CKSPOD, a much smaller standard deviation exists than in KSPOD for any validation case; therefore, as a spatiotemporal emulator, CKSPOD is more desirable than KSPOD.

For all cases, the maximum standard deviation occurs at the LOX inlet area and the recirculation area. This can be explained by the flow physics. The LOX inlet region contains the highest momentum and kinetic energy. Therefore, it is more chaotic and is more difficult to achieve precise and accurate predictions. As it mentioned in Section 4.4.3, the first local maximum of error for liquid-film thickness along the axial direction occurs at the LOX inlet area. Not surprisingly, the maximum standard deviation, or variance of prediction, is found at

this region as well. Downstream of the injector exit, the dynamics become more broadband and no dominant oscillations appear, because there exist strong interactions between the shear layer and recirculation zone generated from vortex breakdown. Hence, it is reasonable to find a strong TKE in the downstream central recirculation area. A strong kinetic energy indicates faster molecules and more disturbances and explains why the maximum standard deviation, or variance, occurs at the downstream central recirculation area in some cases.

Figures 75 to 82 illustrate the predicted TKE $\hat{k}(\mathbf{x}; t)$ at liquid-film thickness at injector exit over the fully developed time-frame of 6-10 ms, along with the 90% two-sided CI band constructed using Theorem 3. At this point, CKSPOD is shown to present the most precise emulated results for Cases A2, B1, and D1. In these three cases, the TKE curves between simulation and CKSPOD are very close. Moreover, the 90% CI band of CKSPOD TKE covers most of the simulation TKE.

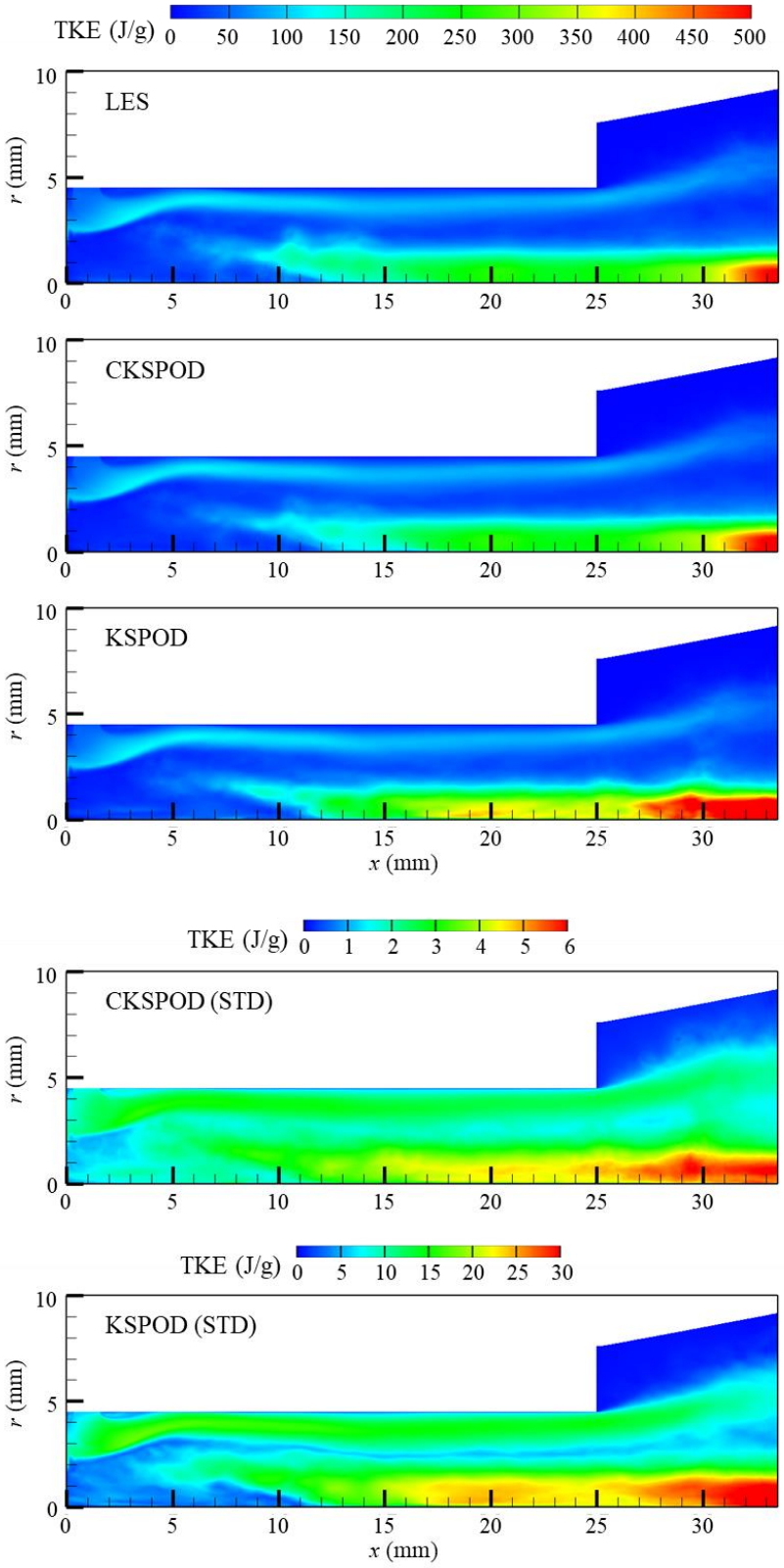

Figure 67—Contour map of temporal averaged TKE and standard deviation for CKSPOD and KSPOD for Case A1

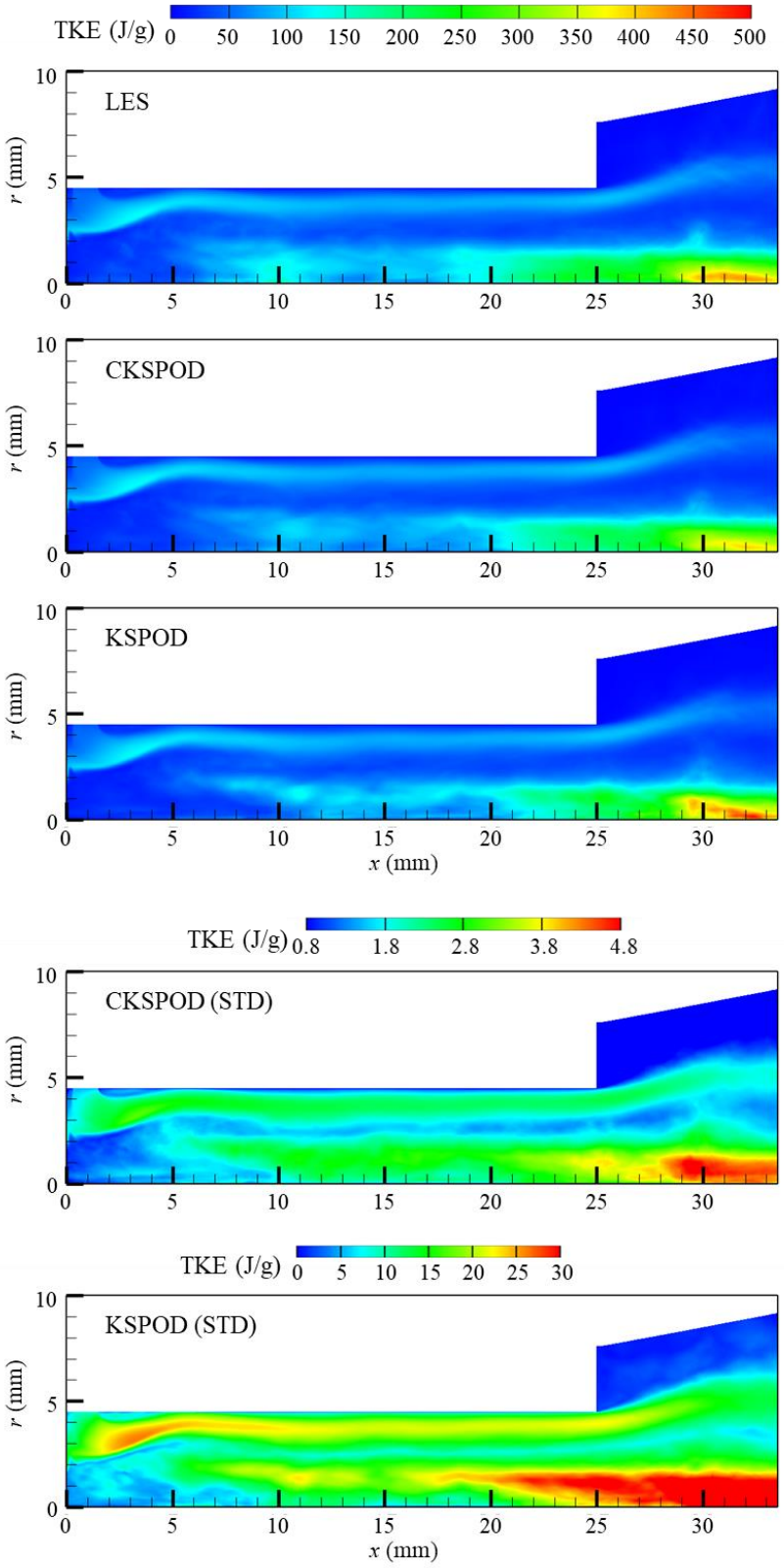

Figure 68—Contour map of temporal averaged TKE and standard deviation for CKSPOD and KSPOD for Case A2

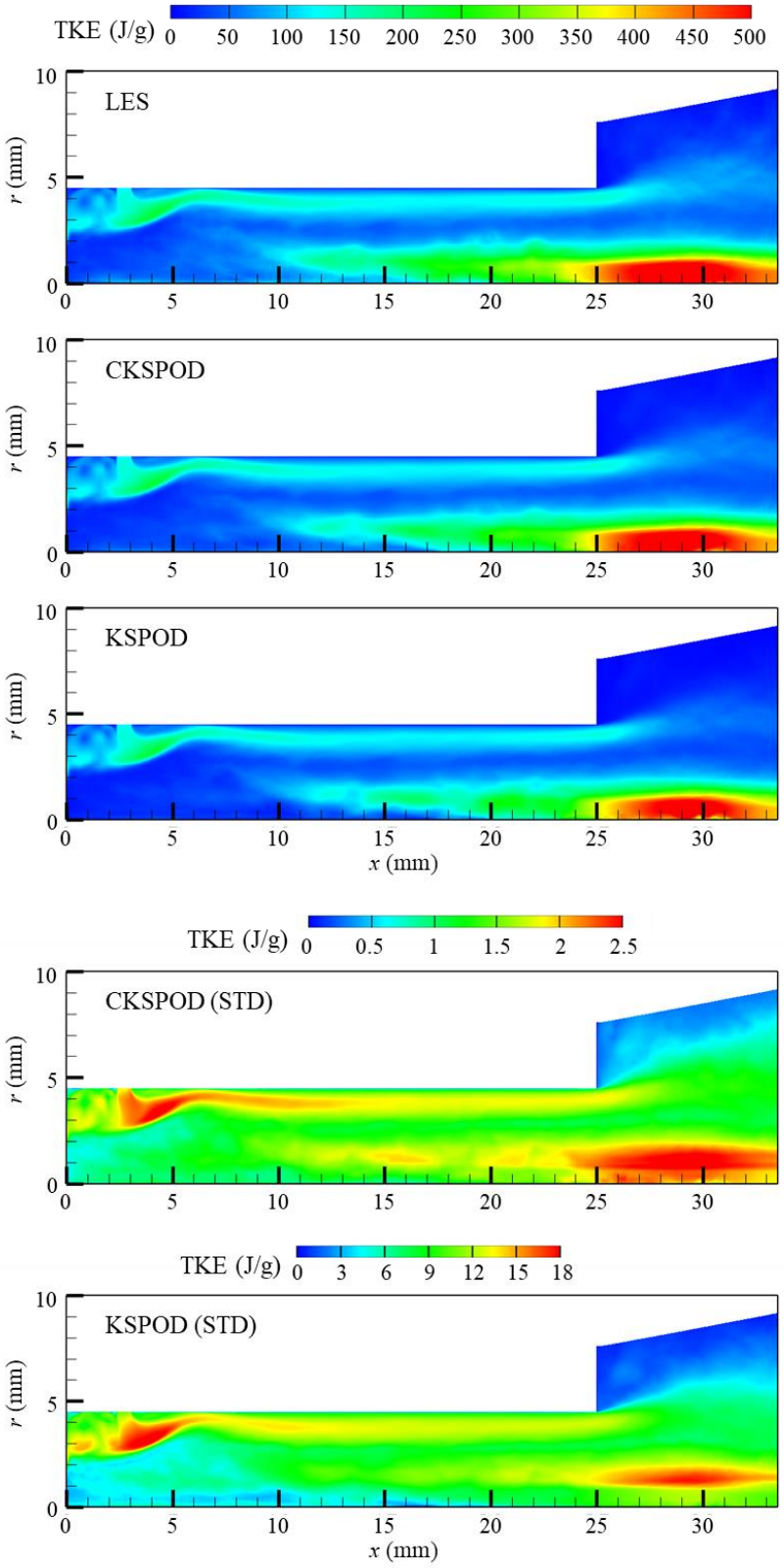

Figure 69—Contour map of temporal averaged TKE and standard deviation for CKSPOD and KSPOD for Case B1

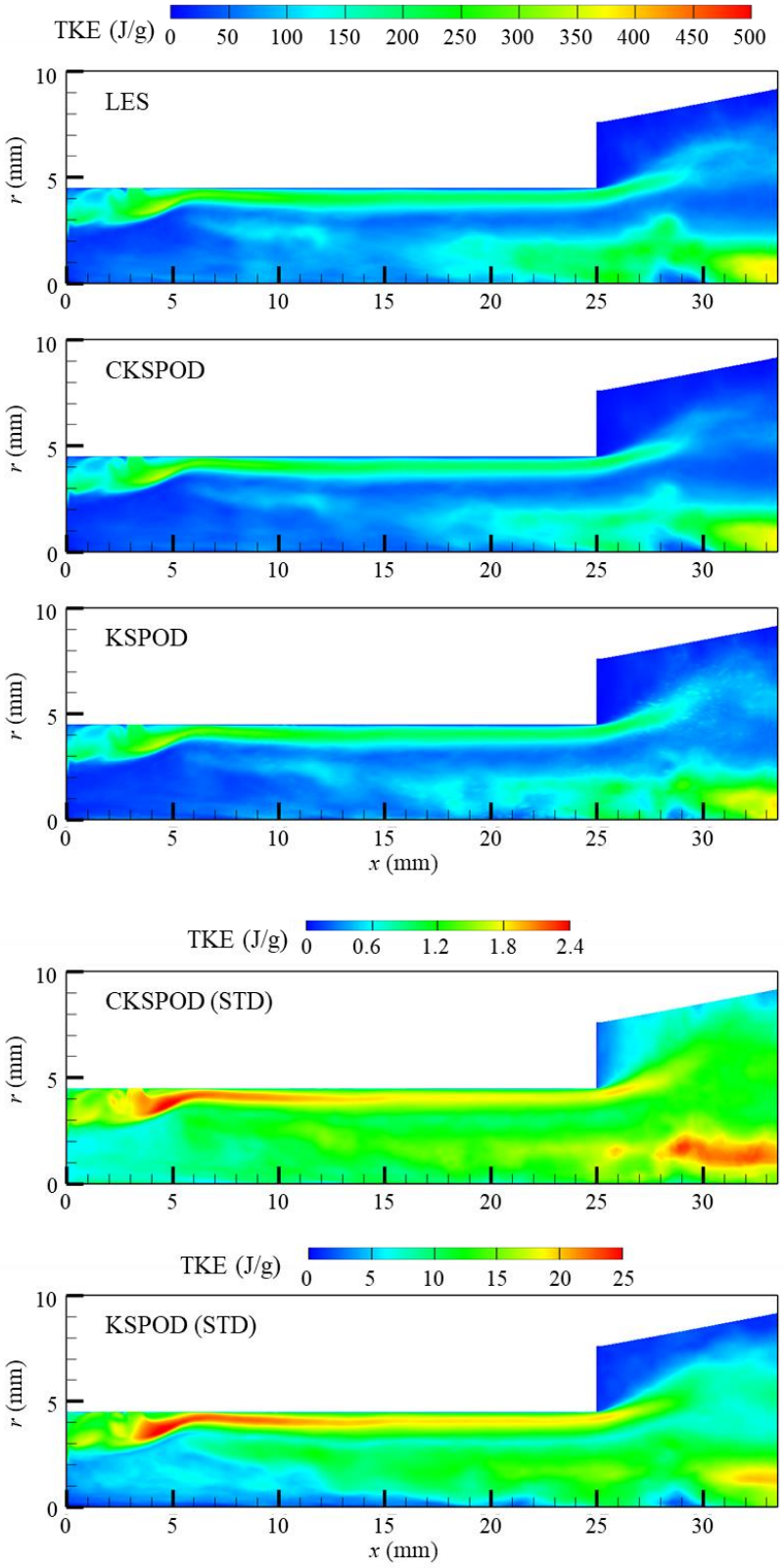

Figure 70—Contour map of temporal averaged TKE and standard deviation for CKSPOD and KSPOD for Case B2

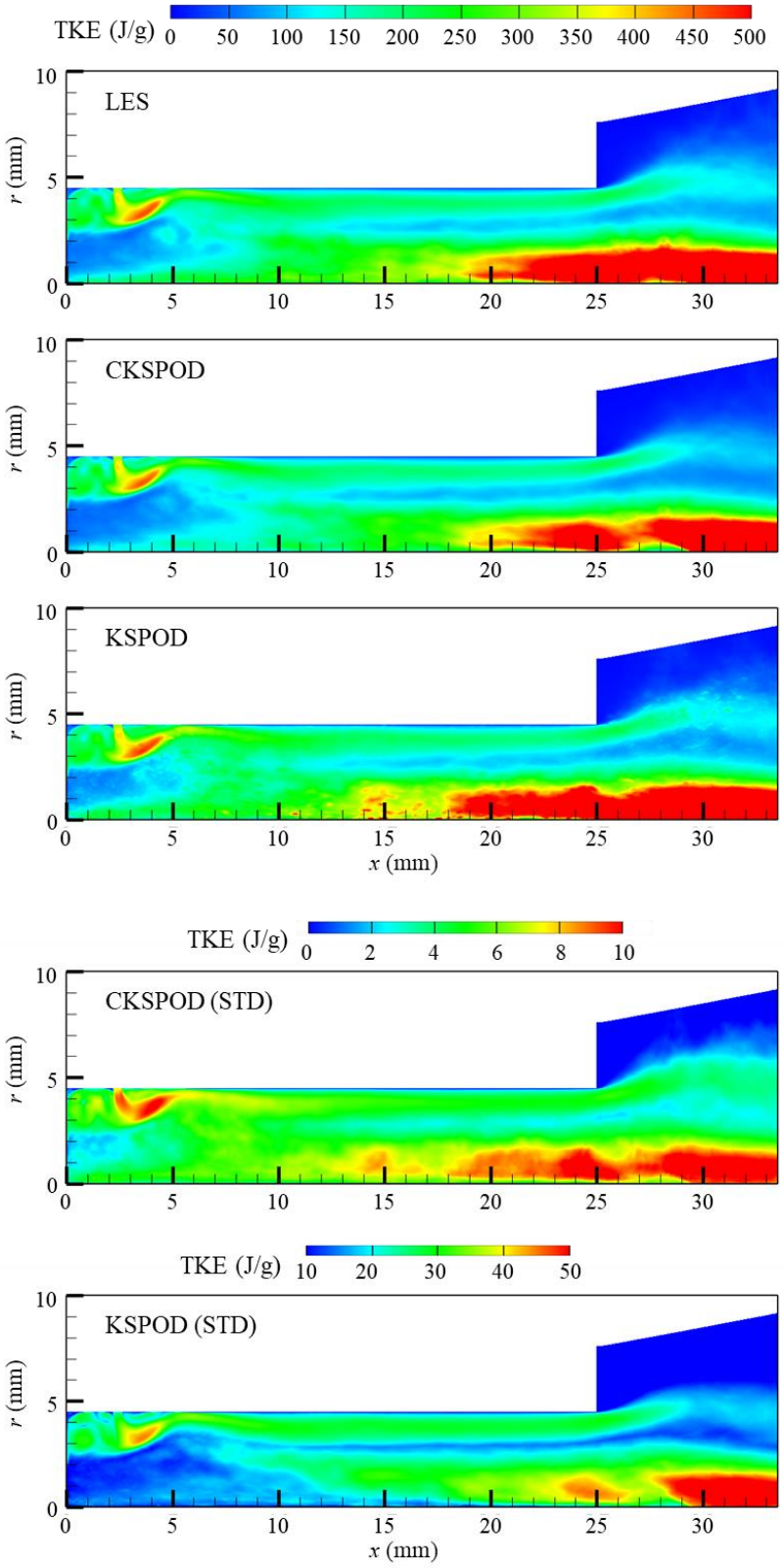

Figure 71—Contour map of temporal averaged TKE and standard deviation for CKSPOD and KSPOD for Case C1

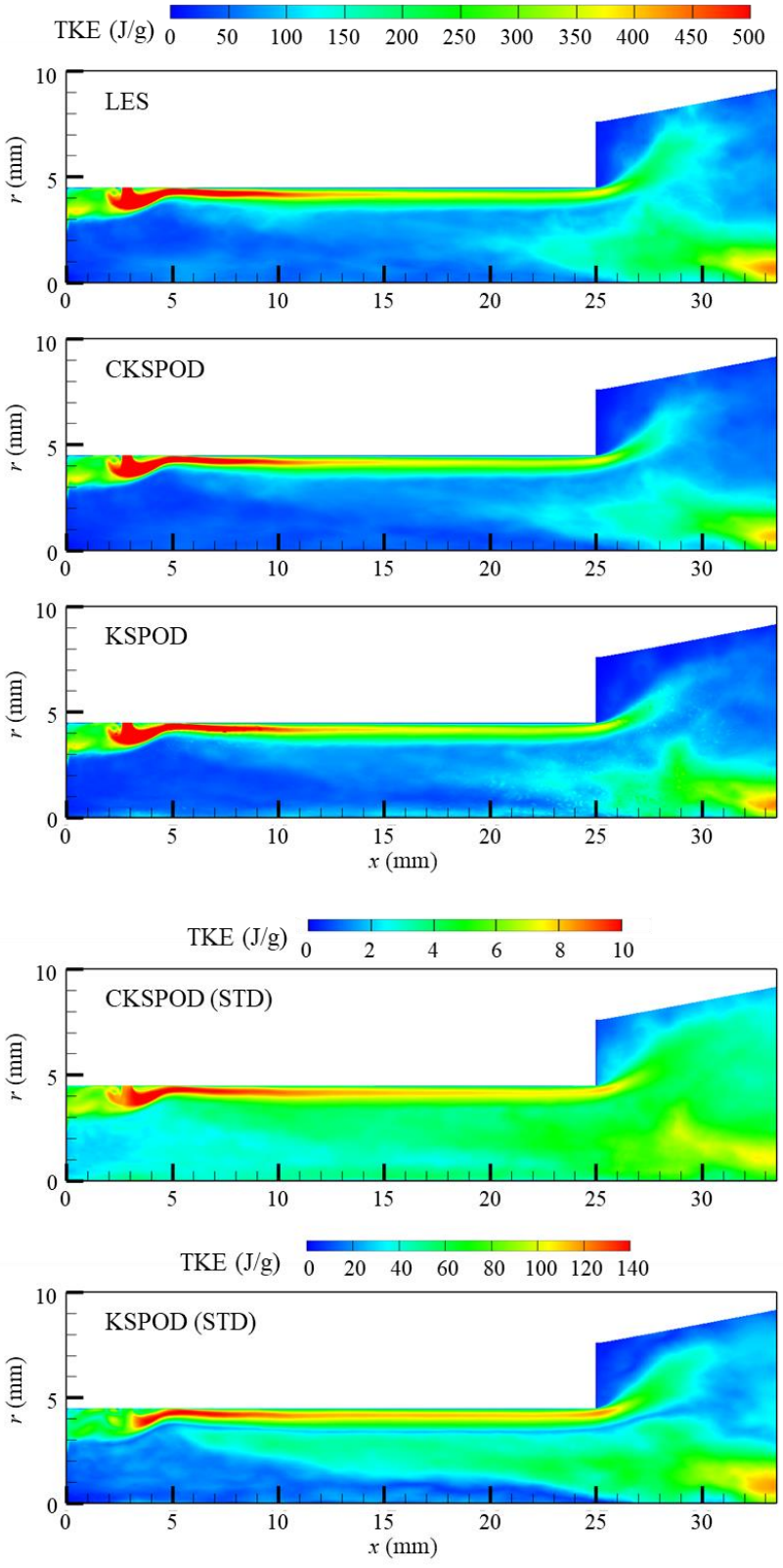

Figure 72—Contour map of temporal averaged TKE and standard deviation for CKSPOD and KSPOD for Case C2

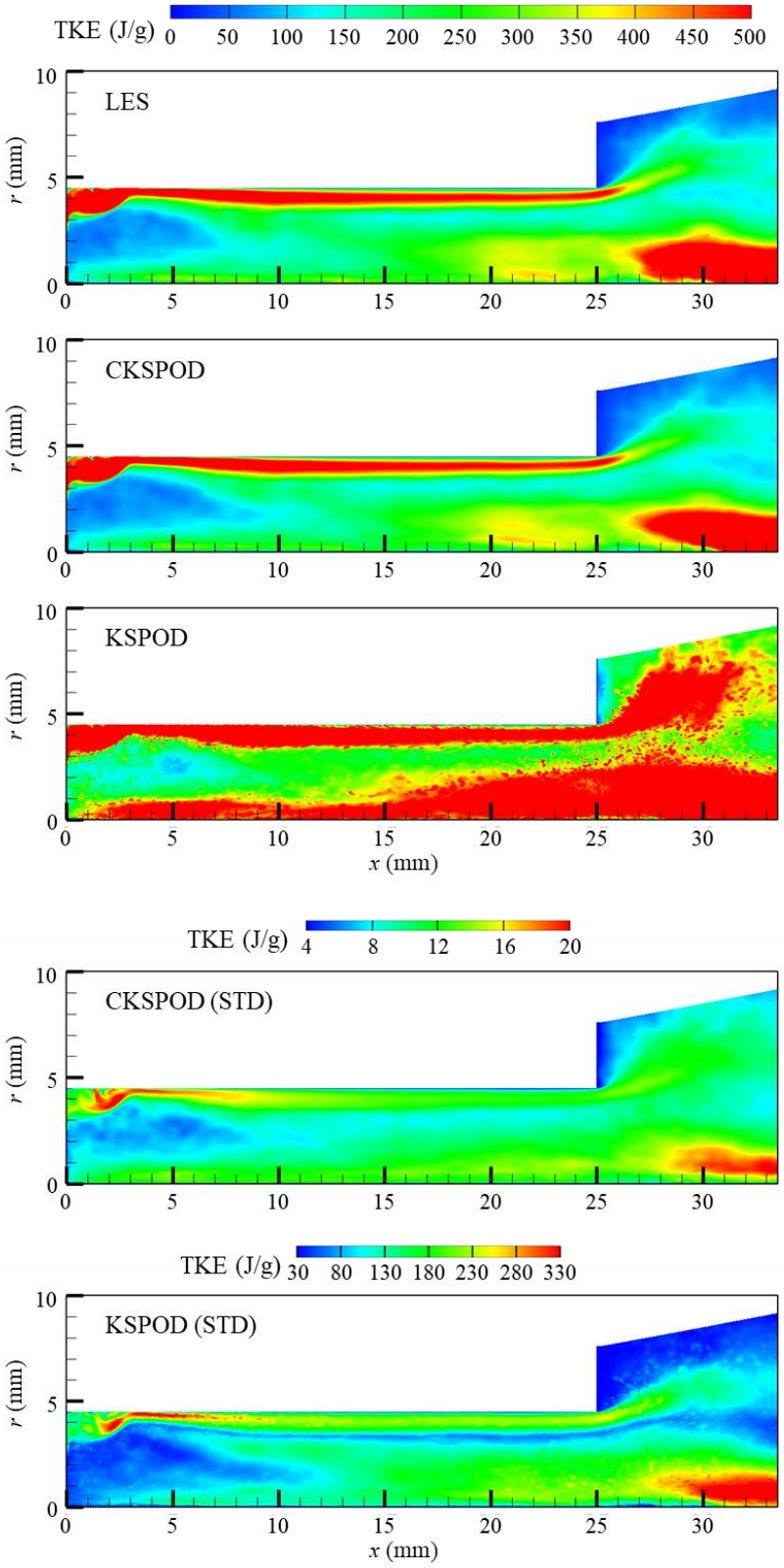

Figure 73—Contour map of temporal averaged TKE and standard deviation for CKSPOD and KSPOD for Case D1

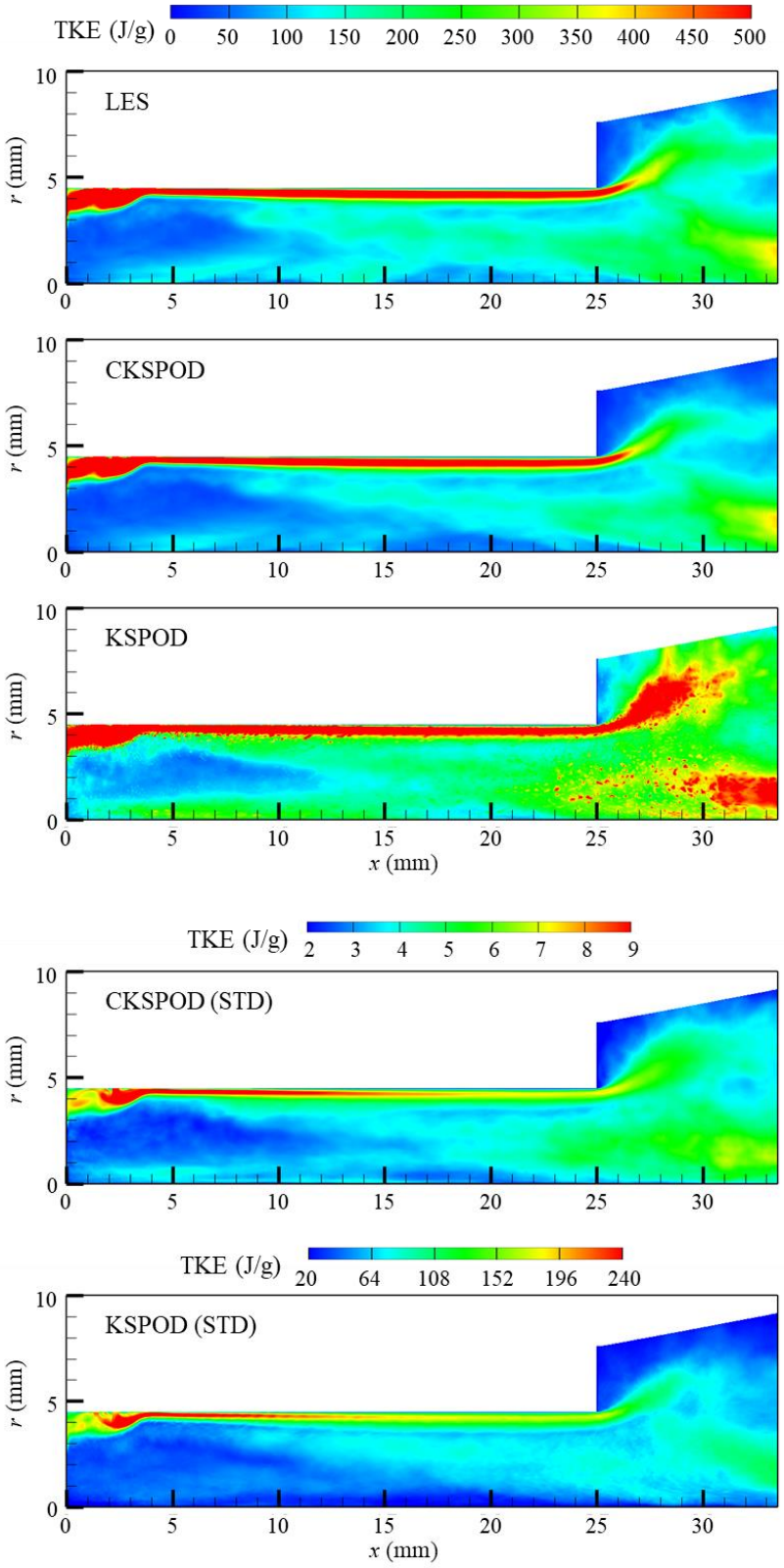

Figure 74—Contour map of temporal averaged TKE and standard deviation for CKSPOD and KSPOD for Case D2

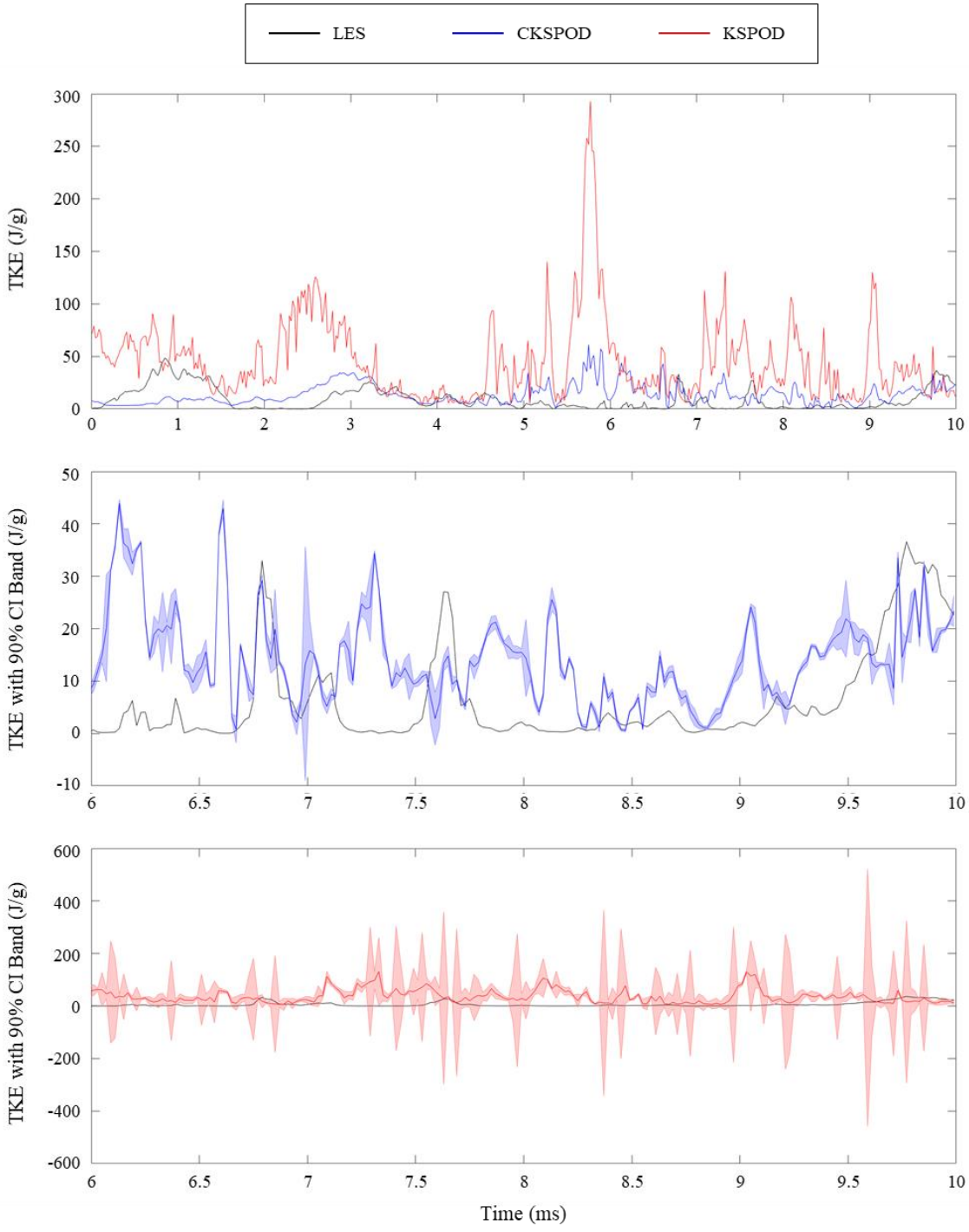

Figure 75—Predicted TKE and 90% CI band for CKSPOD and KSPOD at injector exit of liquid-film thickness point for Case A1

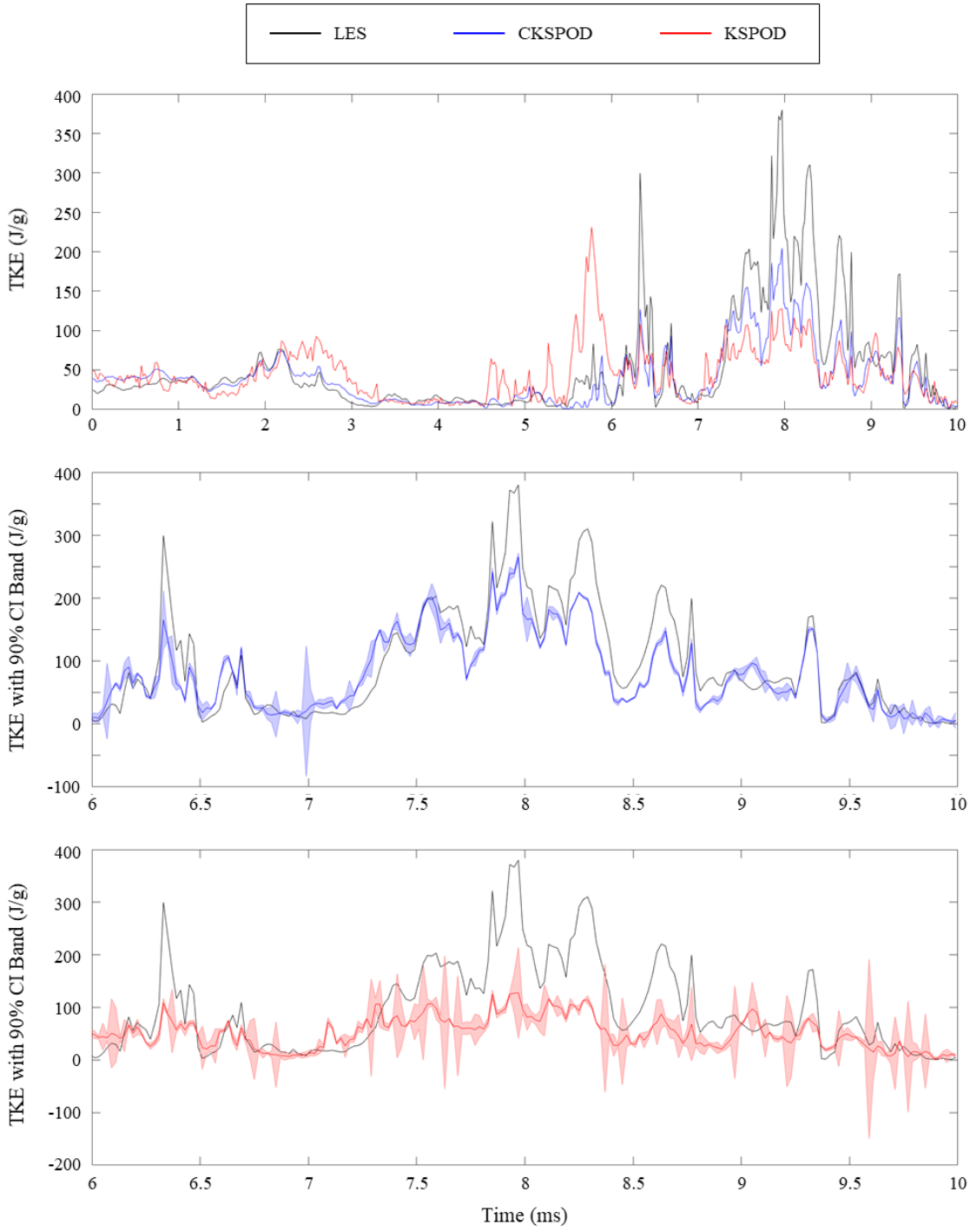

Figure 76—Predicted TKE and 90% confidence interval (CI) band for CKSPOD and KSPOD at injector exit of liquid film thickness point for Case A2

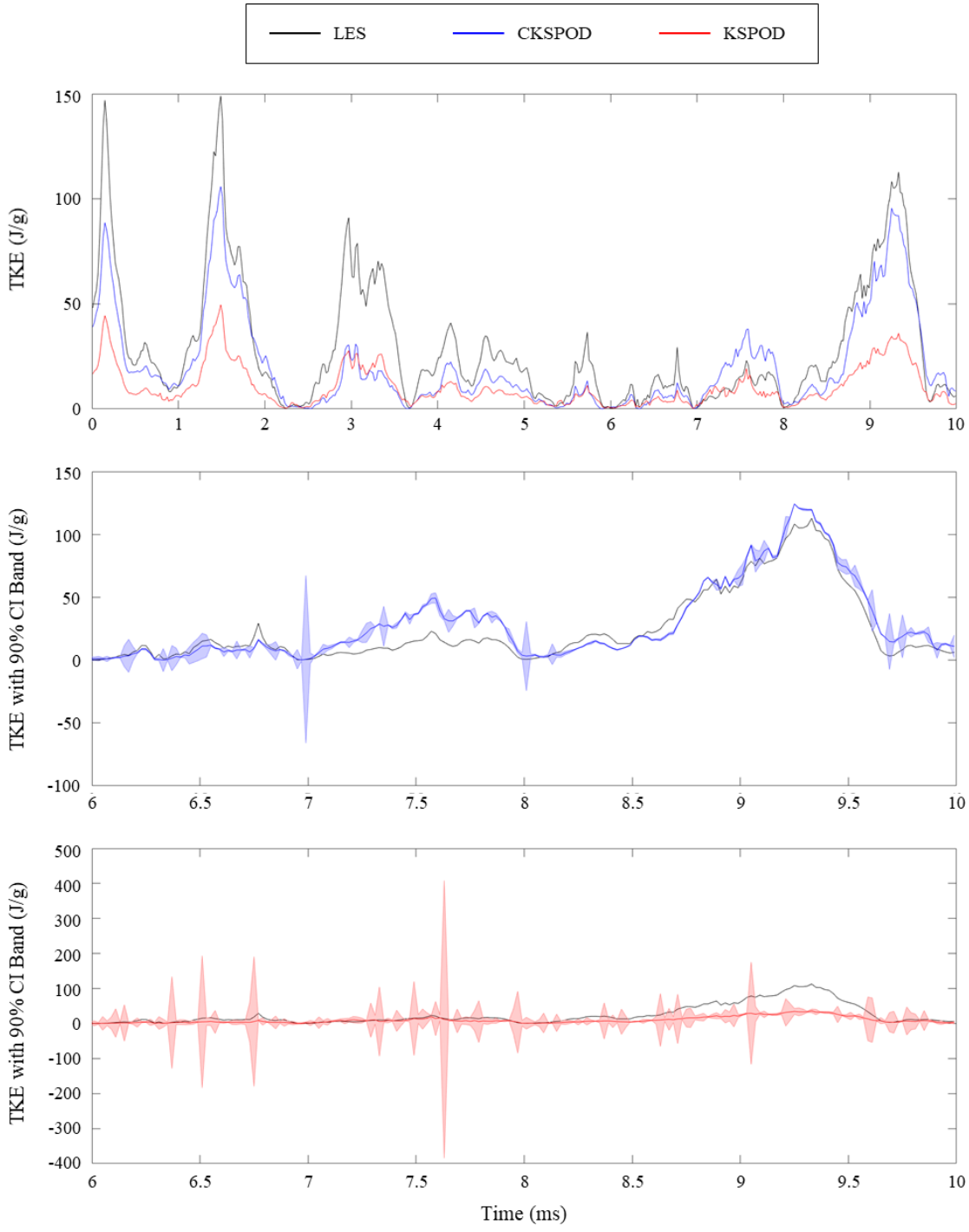

Figure 77—Predicted TKE and 90% CI band for CKSPOD and KSPOD at injector exit of liquid film thickness point for Case B1

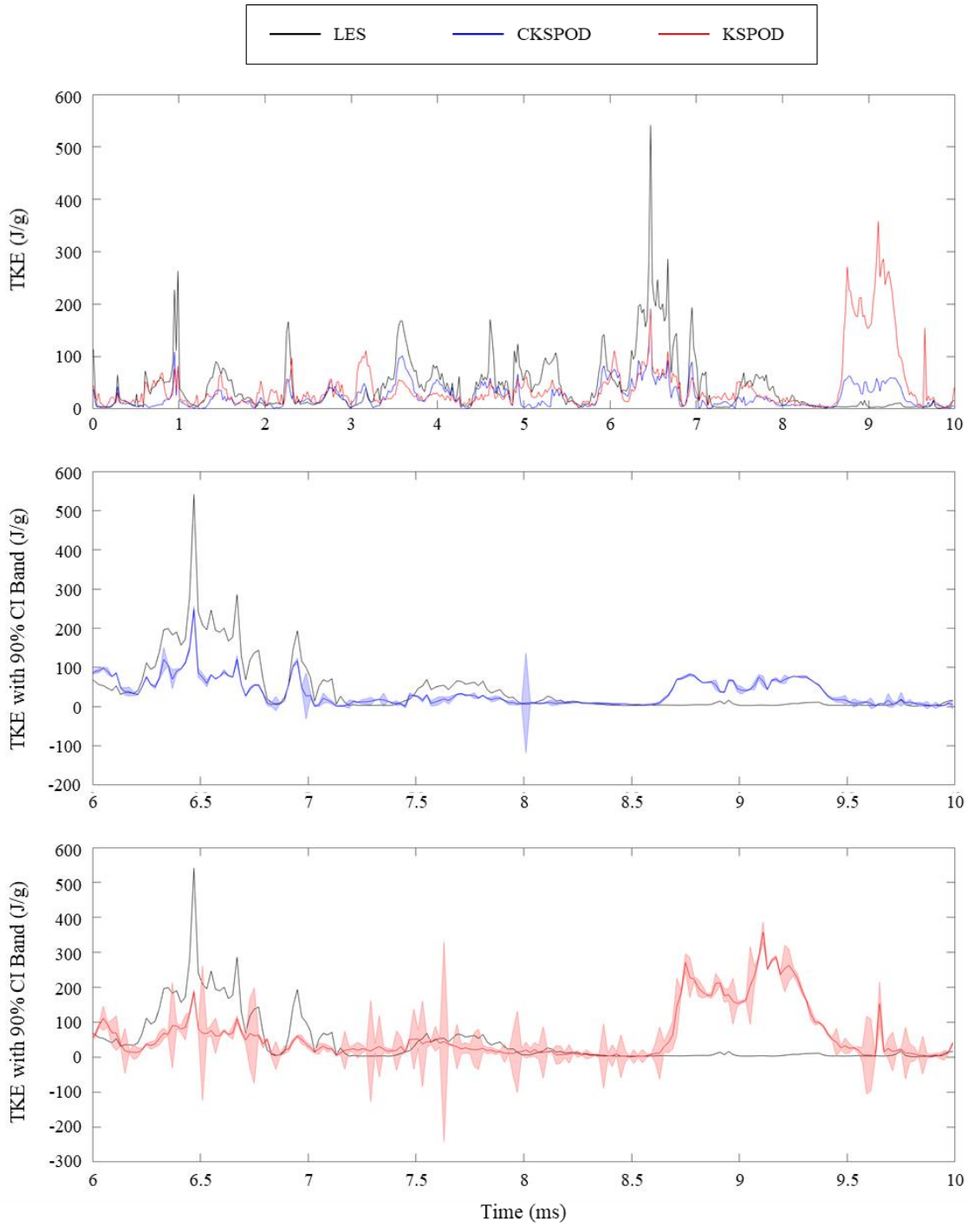

Figure 78—Predicted TKE and 90% CI band for CKSPOD and KSPOD at injector exit of liquid film thickness point for Case B2

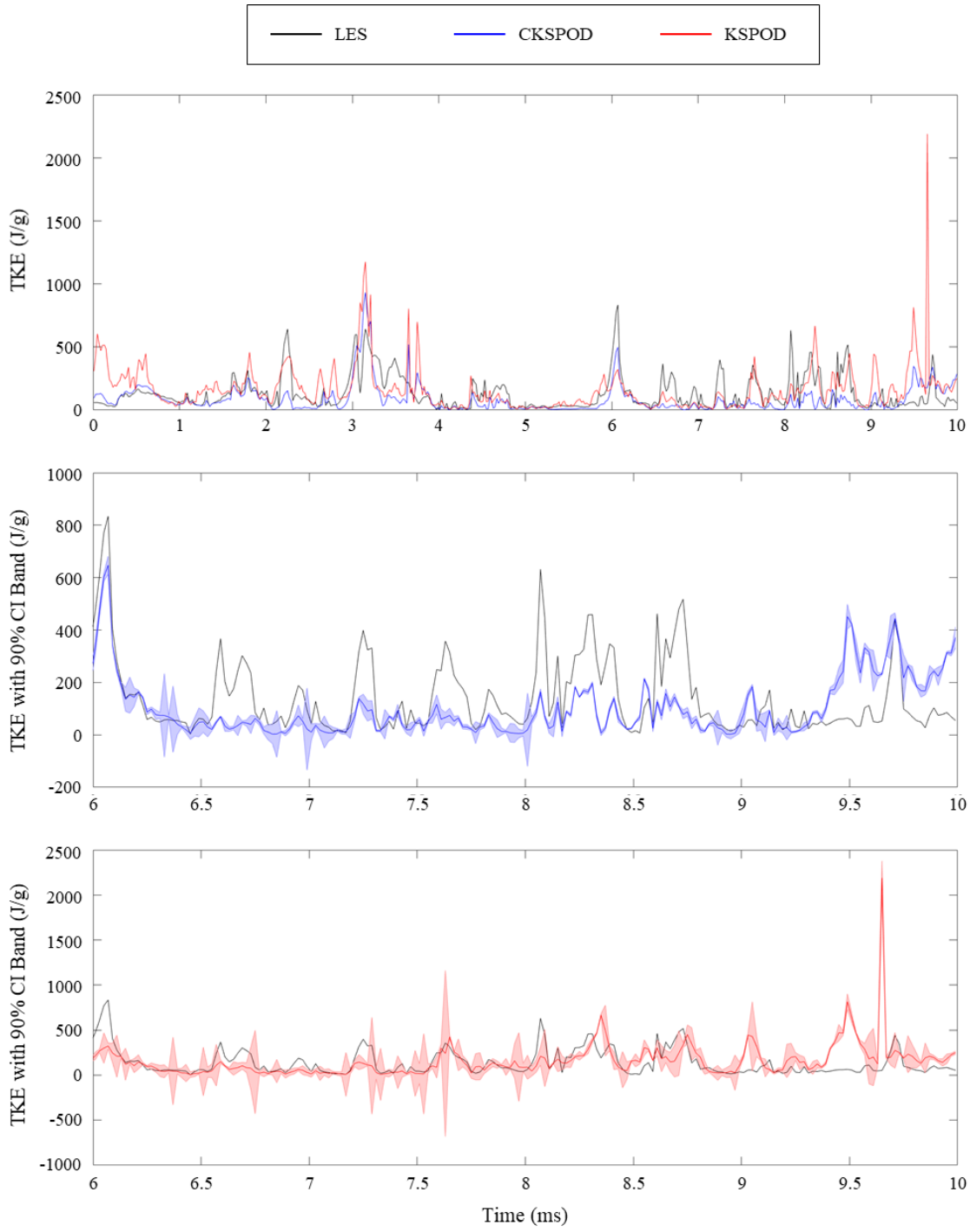

Figure 79—Predicted TKE and 90% CI band for CKSPOD and KSPOD at injector exit of liquid film thickness point for Case C1

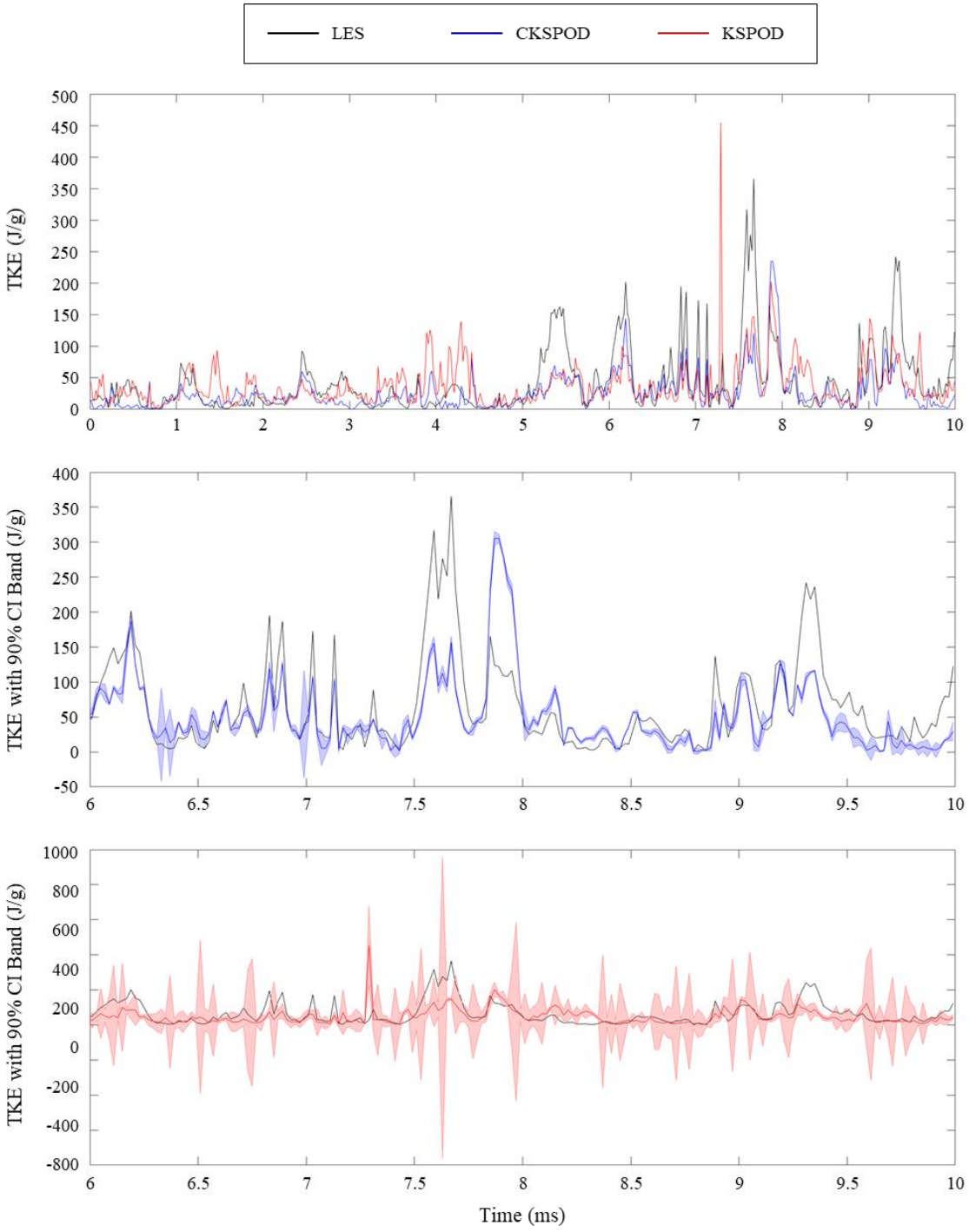

Figure 80—Predicted TKE and 90% CI band for CKSPOD and KSPOD at injector exit of liquid film thickness point for Case C2

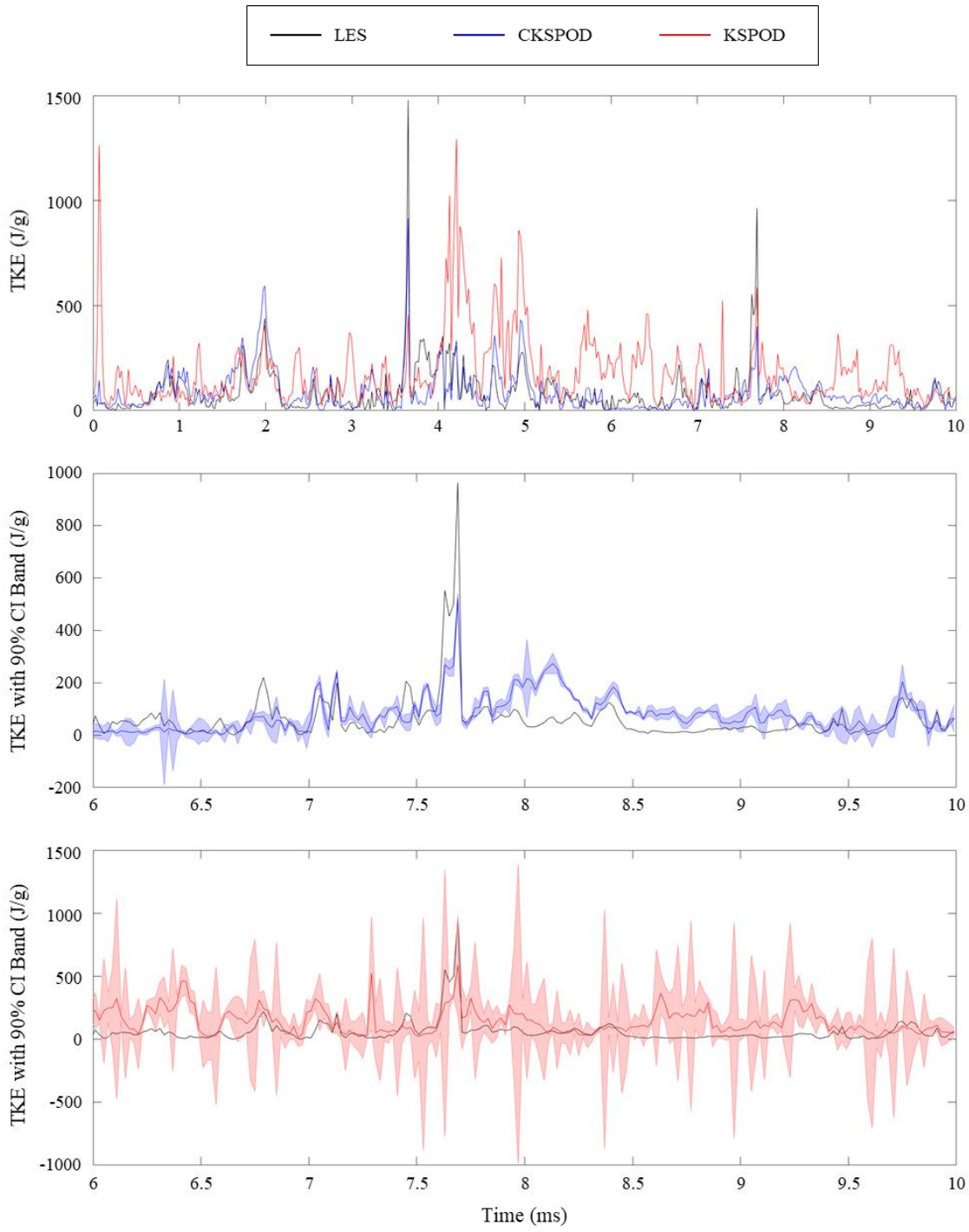

Figure 81—Predicted TKE and 90% CI band for CKSPOD and KSPOD at injector exit of liquid film thickness point for Case D1

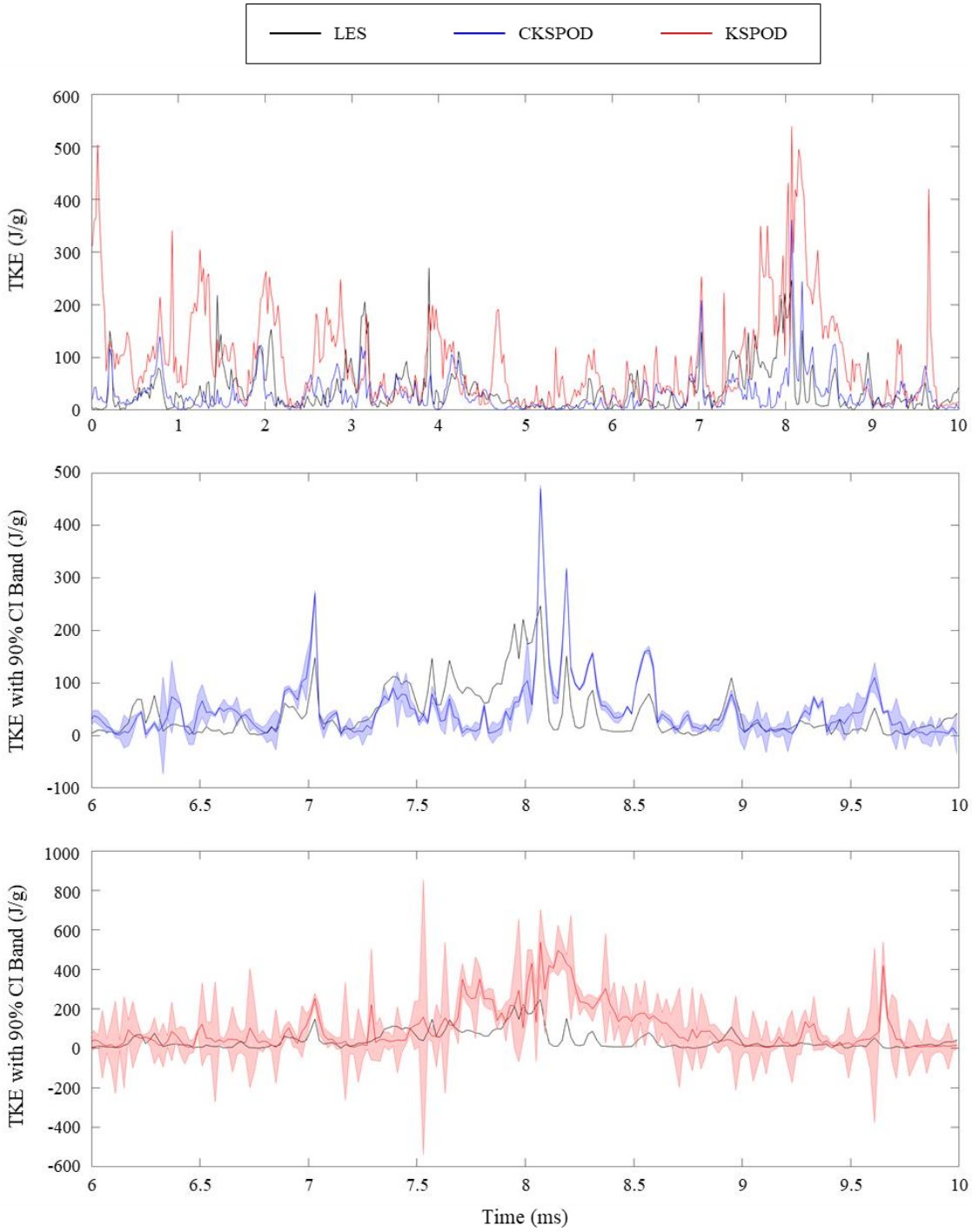

Figure 82—Predicted TKE and 90% CI band for CKSPOD and KSPOD at injector exit of liquid film thickness point for Case D2

4.6 Problems of Emulation for Reacting-Flow Cases

Nonreacting flow dynamics are complex; however, more complex processes exist in the real world. In engines, combustion is a major process that creates reacting flow with more complicated dynamics and chemical reactions than a nonreacting flow dynamics. A high-fidelity simulation for reacting flow at supercritical conditions may use two to three times CPU time than a nonreacting flow simulation.

This section deals with the combustion characteristics of RD-170, a jet-swirl-type bi-propellant injector, which generally includes a gaseous oxidizer jet in the center and a liquid kerosene in the coaxial swirler, as shown in Figure 6. The swirl-induced centrifugal force renders the cold liquid to flow along the annulus wall and thus provide necessary cooling and protection of the injector surface from the hot products. The gaseous core acts as an acoustic resonator to transmit acoustic energy from the combustion chamber to the gas manifold. Similar designs of jet-swirl-type bi-propellant injectors have been widely used in high-pressure, high-thrust combustion chambers of liquid rocket engines, including NK-33, RD-120, and RD-180. The great potential of this injector type has spurred increasing efforts of R&D in many countries.

In this research, the reaction flow training datasets are based on RD-170, a classic gaseous oxygen/kerosene jet-swirl injector, at supercritical conditions. The choice of a different type of injector for reacting flow than for nonreacting flow is made to explore the reliability and application of POD-based emulation models. Also, the reacting-flow cases have lower mesh density but more complex physics characteristics, which provide a better

sample than the nonreacting-flow cases with high mesh resolution to demonstrate the improvement of CKSPOD.

Although jet-swirl injectors have been successfully applied in various propulsion systems, the understanding of the intrinsic mechanisms of propellant injection, mixing, and combustion processes is still limited. It is known that injectors strongly influence the stability characteristics of the combustion system, because the feedback couplings between the combustion chamber and other engine components occur through the injection process [1-2]. The design of an injector is a primarily empirical endeavor that highly relates to design heritage. A type of injector is usually built for a specific application associated with the propellant type and operational conditions. An exhaustive investigation for thorough knowledge of the reaction flow dynamics is needed to improve future injector design and optimization. To that end, a robust emulator may be the key to solving the time-consuming issue of gathering enough flow and flame dynamics information.

Since reacting-flow dynamics are much more complex than nonreacting flow dynamics, the fuel recess length, L_r, is the only design parameter considered in this investigation. The baseline design for L_r is 10.5 mm, and the full size is 16 mm. Hence, by following the 5-10d design rule, the original design matrix uniformly separated L_r by six points including the two ends, which are Cases 1 to 6 in Table 5; Cases 7-10 are added afterwards. Considering different meshes caused by different fuel recess regions among all training cases, both KSPOD-based and CKSPOD-based emulators only adopt the flame developing and intensive combustion areas downstream. However, the results in Figure 83 demonstrate that the six training cases are not sufficient to model a robust emulator.

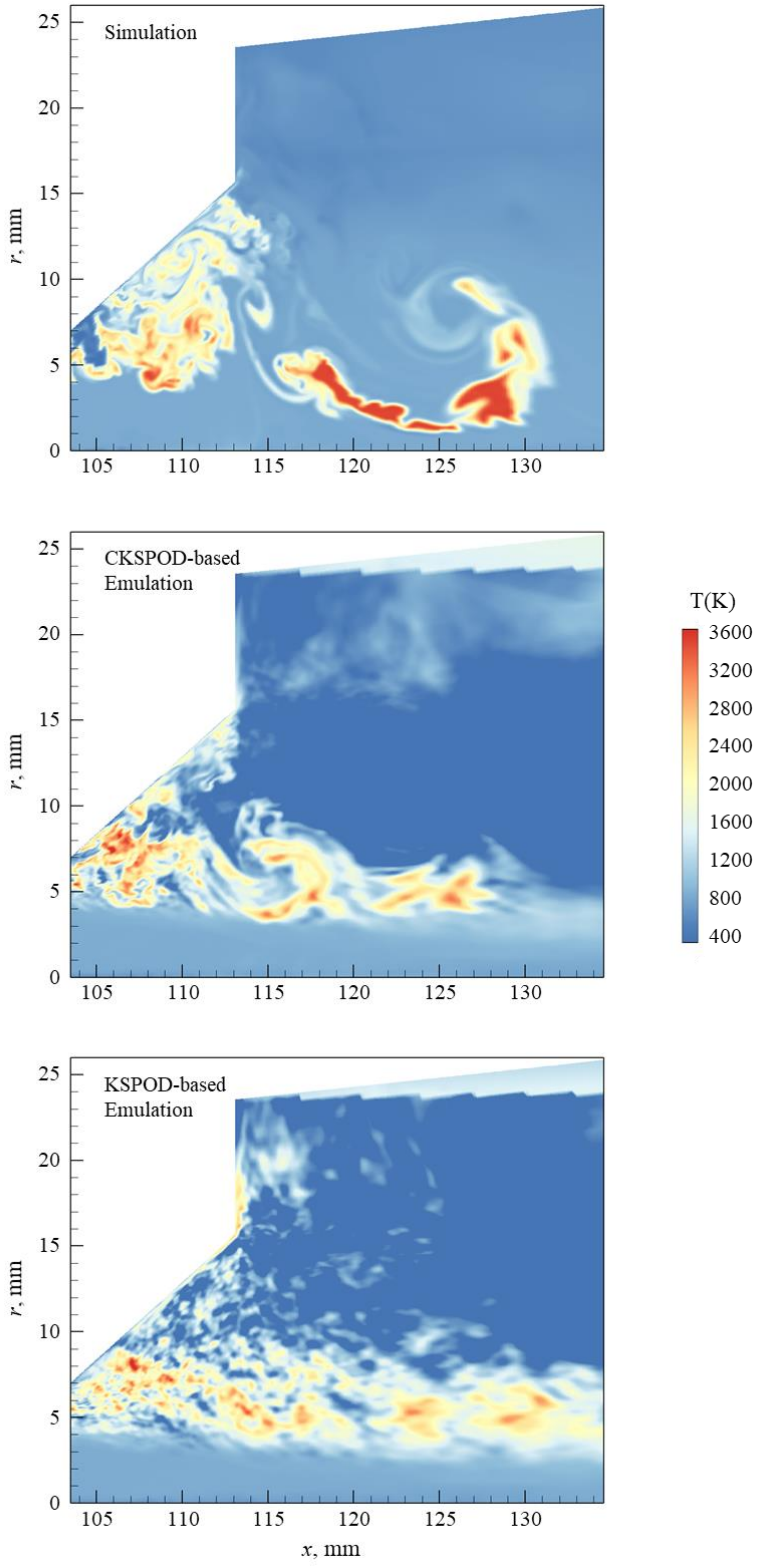

Figure 83—Comparison of temperature field among LES-based simulation, CKSPOD-based emulation, and KSPOD-based emulation at $L_r = 8.75$ mm

Figure 83 shows two emulations for temperature fields by KSPOD and CKSPOD at $t = 0.03$ ms. The new design point is located at $L_r = 5$ (between Cases 4 and 5). The emulations fail to illustrate any clear and reasonable temperature fields demonstrating the flame for either case. These results reveal two important findings: 1) the uncertainty caused by KSPOD is too high, so that KSPOD is unable to emulate a supercritical combustion case and 2) although the design is a linear variation along the recess length, the dynamics variation with the recess length inside this GOX jet is not linear. The flow and flame dynamics are inherently different between Case 1 (fully recessed) and Case 6 (no recess region).

For Case 1, the kerosene is radially injected to the GOX jet with an oxidizer/fuel momentum flux ratio of 3.5, due to a no-shielding design. The kerosene stream must adjust its direction due to the stronger inertial effect of the GOX jet and so flows in two different ways: upstream, with a negative axial velocity, and downstream, driven by the GOX jet. This provides a longer residence time for the interaction of kerosene and GOX than other cases.

The flow and flame dynamics vary gradually between Cases 2 and 5, and their physical characteristics are similar. For Cases 2-3 ($L_r \geq 10.5$ mm, $L_s \leq 5.5$ mm), the distribution of axial velocity is asymmetric, with the peak magnitude below the centerline. The shielding length is too short to allow the kerosene stream to be fully developed. When the shielding length continues to increase, the flow residence time in the swirler increases, leading to a fully developed flow for Cases 4 and 5. The profile of axial velocity becomes asymmetric again, with the peak value shifting above the centerline for the no-recess region in Case 6 ($L_r = 0$). The explanation of this deviation is two-fold: 1) the kerosene stream

spreads upwards due to the sudden expansion in the open taper region as well as the swirl-induced centrifugal force and 2) the expansion of hot products on the surface of GOX and kerosene drives the kerosene to flow along the taper surface.

This study explores the effects of recess length on flow and flame dynamics in depth. The efficiency of propellant mixing and combustion increases with increasing recess length. For a fully recessed injector, the injection of kerosene resembles a jet in crossflow, and two recirculating zones containing kerosene-rich mixture are formed between the injection slit and the headend. A broad flame region is established at the exit of the recess region. For a non-recessed injector, the occurrence of combustion is delayed to the taper region. The flame resides along the taper surface and the faceplate, leaving most oxygen performing convention in downstream without burning.

Overall, a dramatic difference exists between the flow and flame dynamics at the two ends of this DoE. Moreover, a nonlinear design comes with a nonlinear flow and flame dynamics variation. This explains why the emulated results of RD-170 are markedly poor, even with the CKSPOD-based emulator. To improve the emulator, three more points are added between Cases 2 and 5. All ten cases are listed in Table 5, and Case 7 is still applied for validation. Although Cases 8 to 10 are added afterwards and do not follow the DoE rules strictly, these training cases still enhance the performance of CKSPOD emulator (Figure 84).

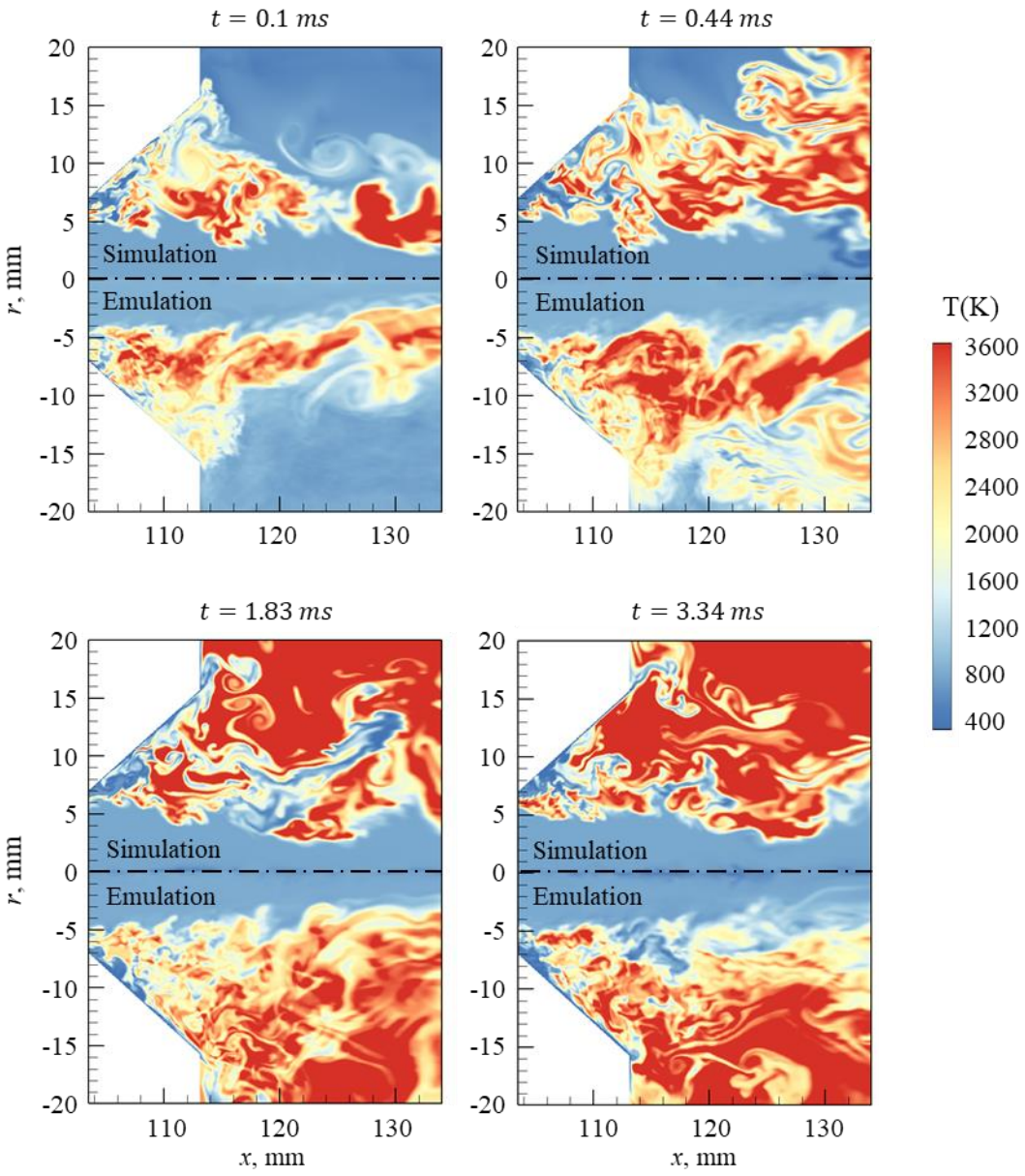

Figure 84—Comparison of temperature field between high-fidelity simulation (LES) and CKSPOD-based low-fidelity emulation for supercritical combustion at $L_r = 8.75$ mm

Figure 84 demonstrates a time-serial comparison of temperature field between simulation and CKSPOD-based emulation at $L_r = 8.75$ mm. Compared with Figure 83, the CKSPOD-based emulations are significantly improved after adding three more training cases within the fuel recess region. It is noted that although the CKSPOD emulator still

cannot construct reacting flowfield as well as it does for nonreacting flow, no time delay exists in emulation for main flow structures. Hence, considering nonlinear output with linear design, the 5-10d rule of thumb design method may not be sufficient to build an emulator for a complex process such as supercritical combustion. Also, one design variable with linear design matrix may not be a sufficient to capture the overall nonlinear flow and flame dynamics variation. A second design variable may need to be considered. Nevertheless, the current database collected from the high-fidelity simulation can be used to study combustion responses of the injection system with various designs in the future work.

CHAPTER 5. CONCLUSION AND FUTURE WORK

The interdisciplinary research develops a serial integrated framework to obtain a surrogate model for a broad-range design space, which encompasses CFD, state-of-the-art statistical methods, machine-learning algorithms, UQ, and data-reduction methods. Using a swirl injector with nonreacting flow with a broad-range design space for training cases as an example, three kriging-based emulation frameworks with physics-driven data-reduction algorithms are proposed and tested to extract the flow physics, reduce the data, and build an efficient, physics-driven emulator: CPOD, KSPOD, and CKSPOD.

The framework of the CPOD research incorporates sensitivity analysis with key design attributes, physics-guided classification of design parameter sets, and flow evolution modeling for efficient design survey. Using a swirl injector for demonstration, the framework of a CPOD-based emulation is applied to extract the flow physics with data reduction and build a physics-driven efficient emulator with a broad-range design space. A sensitivity analysis using Sobol indices and a decision tree is incorporated into the framework to better inform the model. The novelty of this algorithm is the construction of the model through a CPOD covariance via common grid to access data reduction and extraction of common coherent structures. The major contributions of the CPOD work are 1) the use of statistical and machine-learning techniques to quantify the impact of design parameters on important flow physics and 2) the integration of such methods with physics-guided model assumptions to build an efficient surrogate model for flowfield prediction. A fundamental assumption for this model is that the CPOD, the common basis, retains the rich set of physics over varying geometries of the design space. The CPOD model has

successfully captured the simulation results and evaluated performance measurements better than analytical estimations. The emulated flowfield is validated against the time-averaged LES flowfield to demonstrate how the flow features and injector characteristics are preserved by the model. Moreover, this methodology significantly reduces the computational turnaround time required for assessing a new design. While the focus of the CPOD work is on a data-driven analysis and emulation of flow physics, the principle of applying machine-learning techniques with physics-guided assumptions can be applied to any type of engineering application. Overall, this model successfully obtains the prediction accuracy of time-averaged flow features for new swirl injector designs, and the flow dynamics are captured in the form of power spectrum densities. In CPOD work, it also introduces a model for UQ analysis of predictions, providing a metric for model fit. The significantly reduced computation time required for evaluating new design points enables efficient surveying of the design space.

However, to properly predict spatiotemporal evolving flow dynamics, a new emulation technique must be developed; KSPOD, a kernel-smoothed emulation technique that leverages kriging-based weighted functions from the design matrix, is created. As an example, the spatiotemporal flow evolution in a swirl injector is investigated over a wide range of design parameters and operating conditions. The KSPOD-based emulation model is validated against high-fidelity simulation results obtained from LES. The KSPOD-based model not only qualitatively preserves critical physical mechanisms underlying the flow evolution, but also quantitatively captures performance measures over a wide range of temporal and spatial scales of concern. In addition, the model enables effective design

surveys utilizing high-fidelity simulation data, achieving a turnaround time for evaluating new design points that is 42,000 times faster than the original simulation.

Although the KSPOD-based design successfully captures dynamic details of spatiotemporal evolving flow, the prediction results do not entirely synchronically propagate with the corresponding simulation runs. Moreover, the downstream flow structures from emulation are dissimilar to those from the simulation. CKSPOD, an improved emulation model is proposed after CPOD and KSPOD.

Briefly speaking, the CKSPOD technique decomposes all training cases with the same eigenvalues and leverages kriging-based weighted functions from the design matrix. The uniqueness of CKSPOD-based emulation is its use of a transfer matrix to secure all training cases to retain physics information in the same phase after eigen-decomposition. The model not only preserves the fundamental physical mechanisms underlying the flow evolution but also quantitatively captures the dynamics over a wide range of temporal and spatial scales of concern. In addition, the model accounts for the disadvantages of KSPOD, including time delay and some small vague vortex structures from prediction. The CKSPOD-based emulation provides much more accurate and robust predictions than other methods. Although CKSPOD requires more eigen-decomposition information to build the surrogate model than does KSPOD, this new algorithm still achieves 34,000 times faster turnaround time for evaluating new design points than the original simulation.

Although the turn-around time of CKSPOD algorithm is longer than KSPOD, CKSPOD provides much more robust emulation results than KSPOD. The UQ model proposed in Theorem 3 is proved and tested by CPOD. Then, considering that CPOD

performs neither better than KSPOD nor better than CKSPOD in both visualized spatiotemporal flowfield and performance measures, the proposed model is further applied only to KSPOD and CKSPOD for UQ investigation. Both temporal averaged field results and pointwise results show that CKSPOD has much better precision than KSPOD.

Overall, the proposed emulators, especially CKSPOD, provide accurate flow predictions and capture several key metrics for injector performance. In addition, the proposed model offers three appealing features: (1) it furnishes a physically meaningful quantification of spatiotemporal uncertainty, (2) it extracts significant couplings between flow instabilities, and (3) it provides qualitatively and quantitatively accurate predictions for spatiotemporal evolving flow dynamics. A key advantage of all the emulators over existing flow kriging methods is that these models provide accurate predictions using only a fraction of the time required by simulation. This efficiency is very appealing for engineers, since it allows them to fully explore the desired design space and make timely decisions.

To further investigate the capability of these emulators, KSPOD- and CKSPOD-based models are applied to the reaction flow training datasets based on a classic gaseous oxygen/kerosene jet-swirl injector at supercritical conditions. These emulators successfully capture the LES results with high-fidelity accuracy based on some critical assumptions to retain the rich set of physics over varying geometries. Hence, it is important to further explore the limits and capacity of these emulation models. A jet-swirl-type bi-propellant injector including a gaseous oxidizer jet in the center and a liquid kerosene in the coaxial swirler is chosen as a reacting flow example to test the proposed emulators. The current results show that the KSPOD-based emulation model failed to visually predict

spatiotemporal evolving flow with supercritical combustion. The CKSPOD-based emulation model, which has fewer uncertainties on modeling performance than with KSPOD, better visually predicts the same case. Also, no significant time-delay phenomena on flow structure predictions by CKSPOD-based emulator exist. Nevertheless, the emulated results with reacting-flow simulations are still not comparable to the nonreacting-flow simulations. Regarding combustion, the magnitude of emulated results is obviously smaller than those from simulation. In addition, the flow structures in emulated combustion cases are not as detailed as those in nonreacting flow cases. A primary issue for supercritical combustion is that the physics mechanisms do not vary as linearly as the only design parameter. Much work is needed to improve the emulation model for reacting-flow dynamics prediction. With this experience, in the future, it may be possible to develop a more robust emulator with a well-made DoE and a better investigation.

Overall, KSPOD and CKSPOD emulated flowfields are validated against LES nonreacting flowfields with quantitatively and qualitatively strong results; however, these two emulation techniques do not process well with reacting-flow cases. In particular, KSPOD fails to capture a clear evolving flow structure. Moreover, the magnitude of CKSPOD-based emulation results is smaller than the magnitude of related simulation. Hence, to build a time-efficient surrogate model for designs with spatiotemporal dynamics evolution, several factors must be considered:

1. A physics-driven data-reduction process must be considered.
2. The physical mechanism information from each training case must be captured and preserved individually as is the case with KSPOD and CKSPOD.

3. Spatial information and time information should be both emulated, either together or separately.
4. The DoE may vary some design parameters linearly; however, the physics mechanism among all designed experiments may change nonlinearly. Hence, to investigate a process with complex physics mechanism, a smaller range of design space and more than one design parameter must be considered.
5. To study a problem with complex physics mechanism, 5-10*d* design rule may not be sufficient, and more runs of experiments may be needed.

Current results reveal that the proposed emulation models, especially CKSPOD, facilitate effective design surveys with the utilization of high-fidelity simulation data. For conducting further design surveys, an investigation is necessary to improve the emulation model to alleviate the uncertainties of prediction for a very broad-range design space with multiple design parameters.

The originality in this dissertation is that this work offers an innovative design strategy with several emulation algorithms for spatiotemporal evolving flow dynamics, which have never been proposed before. It is the very first time that simulation datasets are used with machine learning techniques to build a robust surrogate model which can emulate spatiotemporal evolving flow dynamics of a new design successfully in short turnaround time with detailed flow structures, highly accurate response predictions, and quantified uncertainties.

In the future, the research should focus on the accuracy of POD analysis of emulated results and the improvement of predictions for cases with complex physical

mechanism. It is essential to validate this model on reaction cases such as supercritical combustion flow. Moreover, the use of artificial neural networks could potentially be employed, if an activation function that properly treats the physics of the problem can be identified. More effective incorporation of physical knowledge for model tuning should also be explored and implemented.

APPENDIX A. KRIGING INTRODUCTION

This section states more details about kriging (Gaussian process regression).

A.1. Basics of Kriging

All kriging estimators are but variants of the basic linear regression estimator $\hat{f}(\boldsymbol{x})$ defined as

$$\hat{f}(\boldsymbol{x}) - m(\boldsymbol{x}) = \sum_{i=i}^{n} \lambda_i [f(\boldsymbol{x}_i) - m(\boldsymbol{x}_i)] \tag{A.1}$$

where $\boldsymbol{x}$ and $\boldsymbol{x}_i$ are location vectors for estimation point and one of the neighboring data points, indexed by i; $m(\boldsymbol{x})$ and $m(\boldsymbol{x}_i)$ are expected values (means) of $f(\boldsymbol{x})$ and $f(\boldsymbol{x}_i)$; λ_i is kriging weight assigned to datum $f(\boldsymbol{x}_i)$ for estimation location $\boldsymbol{x}$; same datum will receive different weight for different estimation location; and n is the total number of data points in local neighborhood used for estimation of $\hat{f}(\boldsymbol{x})$ [100].

$f(\boldsymbol{x})$ is treated as a random field with a trend component, $m(\boldsymbol{x})$, and a residual component, $R(\boldsymbol{x}) = f(\boldsymbol{x}) - m(\boldsymbol{x})$. Kriging estimates residual at $\boldsymbol{x}$ as weighted sum of residuals at surrounding data points. Kriging weights, λ_i, are derived from covariance function or semi-variogram, which should characterize residual component. Distinction between trend and residual somewhat arbitrary; varies with scale.

Eq. (A.1) is the basic form of the kriging estimator. The goal is to determine weights, λ_i, that minimize the variance of the estimator

$$\sigma^2(\boldsymbol{x}) = \mathrm{Var}\left[\hat{f}(\boldsymbol{x}) - f(\boldsymbol{x})\right] \tag{A.2}$$

Under the unbiasedness constraint $\mathrm{E}\left[\hat{f}(\boldsymbol{x}) - f(\boldsymbol{x})\right] = 0$.

The random field $f(\boldsymbol{x})$ is decomposed into residual and trend components, $f(\boldsymbol{x}) = R(\boldsymbol{x}) + m(\boldsymbol{x})$, with the residual component treated as a random field with a stationary mean of 0 and a stationary covariance (i.e., a function of lag, $\boldsymbol{h}$, but not position, $\boldsymbol{x}$):

$$\mathrm{E}[R(\boldsymbol{x})] = 0 \tag{A.3}$$

$$\mathrm{cov}[R(\boldsymbol{x}), R(\boldsymbol{x}+\boldsymbol{h})] = \mathrm{E}[R(\boldsymbol{x}) \cdot R(\boldsymbol{x}+\boldsymbol{h})] = \mathrm{cov}_{\mathrm{R}}[\boldsymbol{h}] \tag{A.4}$$

The residual covariance function is generally derived from the input semi-variogram model, $\mathrm{cov}_{\mathrm{R}}[\boldsymbol{h}] = \mathrm{cov}_{\mathrm{R}}[\boldsymbol{h}] - \gamma(\boldsymbol{h})$. Thus, the semi-variogram we feed to a kriging program should represent the residual component of the variable. The three main kriging variants (i.e., simple kriging, ordinary kriging, and kriging with a trend) differ in their treatments of the trend component, $m(\boldsymbol{x}) = \mathrm{E}[f(\boldsymbol{x})]$.

A.2. Simple Kriging

For simple kriging, the trend component is assumed as a constant and known: $m(\boldsymbol{x}) = \mathrm{E}[f(\boldsymbol{x})] = m$. Therefore,

$$\hat{f}(\boldsymbol{x}^*) = m + \sum_{i=i}^{n} \lambda_i(\boldsymbol{x}^*)[f(\boldsymbol{x}_i) - m] \tag{A.5}$$

This estimation is automatically unbiased, since $\mathrm{E}[f(\boldsymbol{x}) - m] = 0$; hence, $\mathrm{E}\left[\hat{f}(\boldsymbol{x}^*)\right] = m = \mathrm{E}[f(\boldsymbol{x})]$. The estimation error $\hat{f}(\boldsymbol{x}^*) - f(\boldsymbol{x})$ is a linear combination of random variables representing residuals at the data points, $\boldsymbol{x_i}$, and the estimation point, $\boldsymbol{x}^*$:

$$\begin{aligned} \hat{f}(\boldsymbol{x}^*) - f(\boldsymbol{x}^*) &= \left[\hat{f}(\boldsymbol{x}^*) - \boldsymbol{m}\right] - [f(\boldsymbol{x}^*) - \boldsymbol{m}] \\ &= \sum_{i=i}^{n} \lambda_i(\boldsymbol{x}^*)R(\boldsymbol{x}_i) - R(\boldsymbol{x}^*) = R^*(\boldsymbol{x}^*) - R(\boldsymbol{x}^*) \end{aligned} \tag{A.6}$$

Using rules for the variance of a linear combination of random variables, the error variance is then given by

$$\boldsymbol{\sigma_E^2}(\boldsymbol{x}) = \mathbf{Var}[\boldsymbol{R}^*(\boldsymbol{x}^*)] + \mathbf{Var}[\boldsymbol{R}(\boldsymbol{x}^*)] - \mathbf{2cov}[\boldsymbol{R}^*(\boldsymbol{x}^*), \boldsymbol{R}(\boldsymbol{x}^*)]$$

$$= \sum_{i=1}^{n}\sum_{j=i}^{n} \boldsymbol{\lambda_i}(\boldsymbol{x}^*)\boldsymbol{\lambda_j}(\boldsymbol{x}^*)\mathbf{cov_R}\left(\boldsymbol{x_i} - \boldsymbol{x_j}\right) + \mathbf{cov_R}(\mathbf{0}) - \mathbf{2}\sum_{i=i}^{n} \boldsymbol{\lambda_i}(\boldsymbol{x}^*)\mathbf{cov_R}(\boldsymbol{x_i} - \boldsymbol{x}^*)$$

(A.7)

To minimize the error variance, we take the derivation of the above expression with respect to each of the kriging weights and set each derivative to zero. This leads to the following system of equations:

$$\sum_{j=1}^{n} \lambda_j(\boldsymbol{x}^*)\mathrm{cov}_{\mathrm{R}}(\boldsymbol{x}_i - \boldsymbol{x}_j) = \mathrm{cov}_{\mathrm{R}}(\boldsymbol{x}_i - \boldsymbol{x}^*), \quad i = 1, \dots, n \tag{A.8}$$

Because of the constant mean, the covariance function for $f(\boldsymbol{x})$ is the same as that for the residual component, $\mathrm{cov}(\boldsymbol{h}) = \mathrm{cov}_{\mathrm{R}}(\boldsymbol{h})$, so that we can write the simple kriging system directly in terms of $\mathrm{cov}(\boldsymbol{h})$:

$$\sum_{j=1}^{n} \lambda_j(\boldsymbol{x}^*)\mathrm{cov}(\boldsymbol{x}_i - \boldsymbol{x}_j) = \mathrm{cov}(\boldsymbol{x}_i - \boldsymbol{x}^*), \quad i = 1, \dots, n \tag{A.9}$$

This can be written in matrix form as

$$\boldsymbol{\mathcal{K}}\lambda(\boldsymbol{x}^*) = \mathbf{k} \tag{A.10}$$

where $\boldsymbol{\mathcal{K}}$ is the matrix of covariances between data points, with elements $\boldsymbol{\mathcal{K}}_{\boldsymbol{i,j}} = \mathrm{cov}(\boldsymbol{x}_i - \boldsymbol{x}_j)$, $\mathbf{k}$ is the vector covariances between the data points and the estimation point, with elements given by $\mathbf{k} = \mathrm{C}(\boldsymbol{x}_i - \boldsymbol{x}^*)$, and $\lambda(\boldsymbol{x}^*)$ is the vector of simple kriging weights for the surrounding data points. If the covariance model is licit (meaning the underlying semi-variogram model is licit) and no two data points are collocated, then the data covariance matrix is positive definite, and we can solve for the kriging weights using

$$\lambda(\boldsymbol{x}^*) = \boldsymbol{\mathcal{K}}^{-\mathbf{1}}\mathbf{k} \tag{A.11}$$

Once the kriging weights are evaluated, the kriging estimate and the kriging variance can be both computed by substituting the kriging weights into the error variance expression below:

$$\sigma_E^2(\boldsymbol{x}) = \mathrm{C}[0] - \lambda(\boldsymbol{x}^*)\mathbf{k} = \mathrm{C}[0] - \sum_{j=1}^{n} \lambda_j(\boldsymbol{x}^*)\mathrm{cov}(\boldsymbol{x}_i - x^*). \tag{A.12}$$

The simple kriging finds a set of weights for estimating the variable value $f(\boldsymbol{x}^*)$ at point $\boldsymbol{x}^*$ from values at a set of neighboring data points. The weight on each data point generally decreases with increasing distance to that point, in accordance with the decreasing data-to-estimation covariances specified in the right-hand vector, **k**. However, the set of weights is also designed to account for redundancy among the data points, represented in the data point-to-data point covariances in the matrix $\boldsymbol{\mathcal{K}}$. Multiplying **k** by $\boldsymbol{\mathcal{K}^{-1}}$ will downweigh points falling in clusters relative to isolated points at the same distance.

A.3. Ordinary Kriging

For ordinary kriging, rather than assuming that the mean is constant over the entire domain, we assume that it is constant in the local neighborhood of each estimation point, that is $m(\boldsymbol{x_i}) = m(\boldsymbol{x}^*)$ for each nearby data value, $f(\boldsymbol{x}_i)$, that we are using to estimate $f(\boldsymbol{x}^*)$. In this case, the kriging estimator can be written as

$$\hat{f}(\boldsymbol{x}^*) = m(\boldsymbol{x}^*) + \sum_{i=i}^{n} \lambda_i(\boldsymbol{x}^*)\,[f(\boldsymbol{x}_i) - m(\boldsymbol{x}^*)]$$

$$= \sum_{i=i}^{n} \lambda_i(\boldsymbol{x}^*) f(\boldsymbol{x}_i) + \left[1 - \sum_{i=i}^{n} \lambda_i(\boldsymbol{x}^*)\right] m(\boldsymbol{x}^*) \tag{A.13}$$

and we filter the unknown local mean by requiring that the kriging weights sum to 1, leading to an ordinary kriging estimator of

$$\hat{f}(\boldsymbol{x}^*) = \sum_{i=i}^{n} \lambda_i(\boldsymbol{x}^*) f(\boldsymbol{x}_i) \quad \text{with} \quad \sum_{i=i}^{n} \lambda_i(\boldsymbol{x}^*) = 1 \tag{A.14}$$

In order to minimize the error variance subject to the unit-sum constraint on the weights, we actually set up the system minimize the error variance plus an additional term involving a Lagrange parameter, $\mu(\boldsymbol{x}^*)$:

$$\mathcal{L} = \sigma_E^2(\boldsymbol{x}^*) + 2\mu(\boldsymbol{x}^*)\left[1 - \sum_{i=i}^{n} \lambda_i(\boldsymbol{x}^*)\right] \tag{A.15}$$

so that minimization with respect to the Lagrange parameter forces the constraint to be obeyed:

$$\frac{1}{2}\frac{\partial \mathcal{L}}{\partial \mu} = 1 - \sum_{i=i}^{n} \lambda_i(\boldsymbol{x}^*) = 0 \tag{A.16}$$

In this case, the system of equations for the kriging weights turns out to be:

$$\begin{cases} \sum_{j=i}^{n} \lambda_j(\boldsymbol{x}^*)\mathrm{cov}(\boldsymbol{x}_i - \boldsymbol{x}_j) + \mu(\boldsymbol{x}^*) = \mathrm{cov}(\boldsymbol{x}_i - \boldsymbol{x}^*) \\ \sum_{j=i}^{n} \lambda_j(\boldsymbol{x}^*) = 1 \end{cases} \tag{A.17}$$

where $\mathrm{cov}(\boldsymbol{x}_i - \boldsymbol{x}_j)$ is once again the covariance function for the residual component of the variable. In simple kriging, we could equate $\mathrm{C_R}(\boldsymbol{h})$ and $C(\boldsymbol{h})$, the covariance function for the variable itself, due to the assumption of a constant mean. That equality does not hold there, but in practice the substitution is often made anyway, on the assumption that the semi-variogram, from which $\mathrm{C_R}(\boldsymbol{h})$ is derived, effectively filters the influence of large-scale trends in the mean.

In fact, the unit-sum constraint on the weights allows the ordinary kriging system to be stated directly in terms of the semi-variogram (in place of the $\mathrm{C_R}(\boldsymbol{h})$ values above). In a sense, ordinary kriging is the interpolation approach that follows naturally from a semi-variogram analysis, since both tools tend to filter trends in the mean.

Once the kriging weights (and Lagrange parameter) are obtained, the ordinary kriging error variance is given by:

$$\sigma_E^2(\boldsymbol{x}^*) = \mathrm{C}[0] - \sum_{i=1}^{n} \lambda_i(\boldsymbol{x}^*)\mathrm{cov}(\boldsymbol{x}_i - \boldsymbol{x}^*) - \mu(\boldsymbol{x}^*). \tag{A.18}$$

A.4. Kriging with A Trend

Kriging with a trend (the method formerly known as *universal kriging*) is much liked ordinary kriging, except that instead of fitting just a local mean in the neighborhood of the estimation point, we fit a linear or higher-order trend in the (x, y) coordinates of the data points. A local linear (i.e., first-order) trend model would be given by

$$m(\boldsymbol{z}) = m(x, y) = a_0 + a_1 x + a_2 y \tag{A.19}$$

Including such a model in the kriging system involves the same kind of extension as we used for ordinary kriging, with the addition of two more Lagrange parameters and two extra columns and rows in the $\boldsymbol{\mathcal{K}}$ matrix whose (non-zero) elements are the x and y coordinates of the data points. Higher-order trends (quadratic, cubic, etc.) could be handled in the same way, but in practice it is rare to use anything higher than a first-order trend. Ordinary kriging is kriging with a zeroth-order trend model.

If the variable of interest does exhibit a significant trend, a typical approach would be to attempt to estimate a "de-trended" semi-variogram using one of the methods described in the semi-variogram lecture and then feed this into kriging with a first-order trend. However, Goovaerts (1997) warns against this approach and instead recommend performing simple kriging of the residuals from a global trend (with a constant mean of 0) and then adding the kriged residuals back into the global trend.

REFERENCES

[1] MAK, S., SUNG, C.-L., WANG, X., YEH, S.-T., CHANG, Y.-H., JOSEPH, V.R., YANG, V., and WU, C.J., "An efficient surrogate model for emulation and physics extraction of large eddy simulations," JASA. Vol., 2017

[2] YEH, S.-T., WANG, X.S., Chih-Li, MAK, S., CHANG, Y.-H., ZHANG, L., WU, C.-F.J., and YANG, V., "Common Proper Orthogonal Decomposition-Based Spatio-Temporal Emulator for Design Exploration," AIAA J. Vol., 2018 (in press)

[3] CHANG, Y.-H., ZHANG, L., WANG, X., YEH, S.-T., MAK, S., SUNG, C.-L., WU, C., and YANG, V., "Kernel-smoothed proper orthogonal decomposition (KSPOD)-based emulation for prediction of spatiotemporally evolving flow dynamics," arXiv preprint arXiv:1802.08812. Vol., 2018 (submitted)

[4] GUEVARA, H.H., SORIANO, F.H., TUEBKE, A., VEZZANI, A., AMOROSO, S., COAD, A., GKOTSIS, P., and GRASSANO, N., The 2016 EU Industrial R&D Investment Scoreboard. 2017, Joint Research Centre (Seville site).

[5] YANG, V., "Modeling of supercritical vaporization, mixing, and combustion processes in liquid-fueled propulsion systems," Proceedings of the Combustion Institute. Vol. **28**(1): pp. 925-942, 2000

[6] ZONG, N., and YANG*, V., "Cryogenic fluid jets and mixing layers in transcritical and supercritical environments," Combust. Sci. Technol. Vol. **178**(1-3): pp. 193-227, 2006

[7] HUO, H., and YANG, V., "Large-Eddy Simulation of Supercritical Combustion: Model Validation Against Gaseous H 2–O 2 Injector," J. Propul. Power. Vol. **33**(5): pp. 1272-1284, 2017

[8] WANG, X., WANG, Y., and YANG, V., "Geometric effects on liquid oxygen/kerosene bi-swirl injector flow dynamics at supercritical conditions," AIAA J. Vol.: pp. 1-9, 2017

[9] WANG, X., HUO, H., UNNIKRISHNAN, U., and YANG, V., "A systematic approach toward high-fidelity modeling and efficient simulation of supercritical fluid mixing and combustion," Comb. and Flame. Vol., 2017 (submitted)

[10] BAZAROV, V.G., and YANG, V., "Liquid-propellant rocket engine injector dynamics," J. Propul. Power. Vol. **14**(5): pp. 797-806, 1998

[11] ZONG, N., and YANG, V., "Cryogenic fluid dynamics of pressure swirl injectors at supercritical conditions," Phys. Fluids. Vol. **20**(5): pp. 056103, 2008

[12] WANG, X., HUO, H., WANG, Y., and YANG, V., "Comprehensive study of cryogenic fluid dynamics of swirl injectors at supercritical conditions," AIAA J. Vol.: pp. 1-11, 2017

[13] LOEPPKY, J.L., SACKS, J., and WELCH, W.J., "Choosing the Sample Size of a Computer Experiment: A Practical Guide," Technometrics. Vol. **51**(4): pp. 366-376, 2009

[14] CRESSIE, N., "The origins of kriging," Mathematical geology. Vol. **22**(3): pp. 239-252, 1990

[15] BERKOOZ, G., HOLMES, P., and LUMLEY, J.L., "The proper orthogonal decomposition in the analysis of turbulent flows," Annual review of fluid mechanics. Vol. **25**(1): pp. 539-575, 1993
[16] FISHER, R.A., Statistical methods for research workers. 1925: Genesis Publishing Pvt Ltd.
[17] FISHER, R.A., The design of experiments. 1937: Oliver And Boyd; Edinburgh; London.
[18] BOX, G.E., and WILSON, K.B., On the experimental attainment of optimum conditions, in *Breakthroughs in statistics*. 1992, Springer. p. 270-310.
[19] DEMING, W.E., and EDWARDS, D.W., Quality, productivity, and competitive position. Vol. 183. 1982: Massachusetts Institute of Technology, Center for advanced engineering study Cambridge, MA.
[20] ISHIKAWA, K., What is total quality control? The Japanese way. 1985: Prentice Hall.
[21] TAGUCHI, G., CHOWDHURY, S., and WU, Y., Taguchi's quality engineering handbook. Vol. 1736. 2005: Wiley Online Library.
[22] TAGUCHI, G., Introduction to quality engineering: designing quality into products and processes. 1986.
[23] PANDE, P.S., NEUMAN, R.P., and CAVANAGH, R.R., The six sigma way: How GE, Motorola, and other top companies are honing their performance. 2000: McGraw-Hill (New York).
[24] PLACKETT, R.L., and BURMAN, J.P., "The design of optimum multifactorial experiments," Biometrika. Vol. **33**(4): pp. 305-325, 1946
[25] TAGUCHI, G., and KONISHI, S., Orthogonal arrays and linear graphs: tools for quality engineering. 1987: American Supplier Institute.
[26] RASMUSSEN, C.E., Gaussian processes in machine learning, in *Advanced lectures on machine learning*. 2004, Springer. p. 63-71.
[27] KRIGE, D.G., Two-dimensional weighted moving average trend surfaces for ore evaluation. 1966: South African Institute of Mining and Metallurgy Johannesburg.
[28] KRIGE, D.G., "A statistical approach to some basic mine valuation problems on the Witwatersrand," Journal of the Southern African Institute of Mining and Metallurgy. Vol. **52**(6): pp. 119-139, 1951
[29] MATHERON, G., Les variables régionalisées et leur estimation: une application de la théorie des fonctions aléatoires aux sciences de la nature. 1965, Paris: Masson et CIE.
[30] MATHERON, G., "Principles of geostatistics," Econ. Geol. Vol. **58**(8): pp. 1246-1266, 1963
[31] WEBSTER, R., and BURGESS, T., "Optimal interpolation and isarithmic mapping of soil properties III changing drift and universal kriging," European Journal of Soil Science. Vol. **31**(3): pp. 505-524, 1980
[32] DE MARSILY, G., and AHMED, S., "Application of kriging techniques in groundwater hydrology," Journal of the Geological Society of India. Vol. **29**(1): pp. 57-82, 1987

[33] GAJEM, Y., WARRICK, A., and MYERS, D., "Spatial Dependence of Physical Properties of a Typic Torrifluvent Soil 1," Soil Science Society of America Journal. Vol. **45**(4): pp. 709-715, 1981
[34] MCBRATNEY, A.B., WEBSTER, R., MCLAREN, R.G., and SPIERS, R.B., "Regional variation of extractable copper and cobalt in the topsoil of south-east Scotland," Agronomie. Vol. **2**(10): pp. 969-982, 1982
[35] VAUCLIN, M., VIEIRA, S., VACHAUD, G., and NIELSEN, D., "The Use of Cokriging with Limited Field Soil Observations 1," Soil Science Society of America Journal. Vol. **47**(2): pp. 175-184, 1983
[36] RUSSO, D., "Design of an Optimal Sampling Network for Estimating the Variogram 1," Soil Science Society of America Journal. Vol. **48**(4): pp. 708-716, 1984
[37] OLIVER, M.A., and WEBSTER, R., "Kriging: a method of interpolation for geographical information systems," International Journal of Geographical Information System. Vol. **4**(3): pp. 313-332, 1990
[38] SULLIVAN, T.J., Introduction to uncertainty quantification. Vol. 63. 2015: Springer.
[39] JOSEPH, V.R., GUL, E., and BA, S., "Maximum projection designs for computer experiments," Biometrika. Vol. **102**(2): pp. 371-380, 2015
[40] BONNET, J.P., COLE, D.R., DELVILLE, J., GLAUSER, M.N., and UKEILEY, L.S., "Stochastic estimation and proper orthogonal decomposition: complementary techniques for identifying structure," Exp. Fluids. Vol. **17**(5): pp. 307-314, 1994
[41] NOACK, B.R., AFANASIEV, K., MORZYNSKI, M., TADMOR, G., and THIELE, F., "A hierarchy of low-dimensional models for the transient and post-transient cylinder wake," J. Fluid Mech. Vol. **497**: pp. 335-363, 2003
[42] LUMLEY, J.L., Coherent structures in turbulence, in *Transition and turbulence*. 1981, Elsevier. p. 215-242.
[43] MENG, H., and YANG, V., "A unified treatment of general fluid thermodynamics and its application to a preconditioning scheme," J Comput Phys. Vol. **189**(1): pp. 277-304, 2003
[44] ZONG, N., and YANG, V., "An efficient preconditioning scheme for real-fluid mixtures using primitive pressure–temperature variables," Int. J. Comput. Fluid Dyn. Vol. **21**(5-6): pp. 217-230, 2007
[45] HSIEH, S.-Y., and YANG, V., "A preconditioned flux-differencing scheme for chemically reacting flows at all Mach numbers," Int. J. Comput. Fluid Dyn. Vol. **8**(1): pp. 31-49, 1997
[46] OEFELEIN, J.C., and YANG, V., "Modeling High-Pressure Mixing and Combustion Processes in Liquid Rocket Engines," J. Propul. Power. Vol. **14**(5): pp. 843-857, 1998
[47] ERLEBACHER, G., HUSSAINI, M., SPEZIALE, C., and ZANG, T.A., "Toward the large-eddy simulation of compressible turbulent flows," J. Fluid Mech. Vol. **238**: pp. 155-185, 1992
[48] TAKAHASHI, S., "Preparation of a generalized chart for the diffusion coefficients of gases at high pressures," J. Chem. Eng. Jpn. Vol. **7**(6): pp. 417-420, 1975

[49] ZONG, N., MENG, H., HSIEH, S.-Y., and YANG, V., "A numerical study of cryogenic fluid injection and mixing under supercritical conditions," Phys. Fluids. Vol. **16**(12): pp. 4248-4261, 2004
[50] HUO, H., WANG, X., and YANG, V., "A general study of counterflow diffusion flames at subcritical and supercritical conditions: Oxygen/hydrogen mixtures," Comb. and Flame. Vol. **161**(12): pp. 3040-3050, 2014
[51] WANG, X., HUO, H., and YANG, V., "Counterflow Diffusion Flames of Oxygen and N-Alkane Hydrocarbons (CH4-C16H34) at Subcritical and Supercritical Conditions," Combust. Sci. Technol. Vol. **187**(1-2): pp. 60-82, 2015
[52] POINSOT, T., and VEYNANTE, D., Theoretical and numerical combustion. 2005: RT Edwards, Inc.
[53] DAGAUT, P., EL BAKALI, A., and RISTORI, A., "The combustion of kerosene: Experimental results and kinetic modelling using 1-to 3-component surrogate model fuels," Fuel. Vol. **85**(7-8): pp. 944-956, 2006
[54] WANG, Q.-D., FANG, Y.-M., WANG, F., and LI, X.-Y., "Skeletal mechanism generation for high-temperature oxidation of kerosene surrogates," Comb. and Flame. Vol. **159**(1): pp. 91-102, 2012
[55] SWANSON, R.C., and TURKEL, E., "On central-difference and upwind schemes," J Comput Phys. Vol. **101**(2): pp. 292-306, 1992
[56] CHEN, X., and YANG, V., "Effect of ambient pressure on liquid swirl injector flow dynamics," Phys. Fluids. Vol. **26**(10): pp. 102104, 2014
[57] LI, H.-G., ZONG, N., LU, X.-Y., and YANG, V., "A consistent characteristic boundary condition for general fluid mixture and its implementation in a preconditioning scheme," Advances in Applied Mathematics and Mechanics. Vol. **4**(1): pp. 72-92, 2012
[58] QIAN, P.Z., "Sliced Latin hypercube designs," JASA. Vol. **107**(497): pp. 393-399, 2012
[59] XIONG, F., XIONG, Y., CHEN, W., and YANG, S., "Optimizing Latin hypercube design for sequential sampling of computer experiments," Eng. Optimiz. Vol. **41**(8): pp. 793-810, 2009
[60] LUMLEY, J.L., "The structure of inhomogeneous turbulent flows," Atmospheric turbulence and radio wave propagation. Vol. **790**: pp. 166-178, 1967
[61] STOKES, G.G., On the effect of the internal friction of fluids on the motion of pendulums. Vol. 9. 1851: Pitt Press Cambridge.
[62] POPE, S.B., Turbulent flows. 2001, IOP Publishing.
[63] LOøVE, M., Probability Theory; Foundations, Random Sequences. 1955: D. Van Nostrand Co.
[64] XIAO, M., BREITKOPF, P., COELHO, R.F., KNOPF-LENOIR, C., SIDORKIEWICZ, M., and VILLON, P., "Model reduction by CPOD and Kriging," Structural and multidisciplinary optimization. Vol. **41**(4): pp. 555-574, 2010
[65] JOSEPH, V.R., and KANG, L., "Regression-based inverse distance weighting with applications to computer experiments," Technometrics. Vol. **53**(3): pp. 254-265, 2011

[66] DEXTER, C.E., FISHER, M.F., HULKA, J.R., DENISOV, K.P., SHIBANOV, A.A., and AGARKOV, A.F., Scaling techniques for design, development, and test. 2004, AIAA Reston, VA. p. 553-600.
[67] SANTNER, T.J., WILLIAMS, B.J., and NOTZ, W.I., The design and analysis of computer experiments. 2013: Springer Science & Business Media.
[68] CASELLA, G., and BERGER, R.L., Statistical inference. Vol. 2. 2002: Duxbury Pacific Grove, CA.
[69] LIU, D.C., and NOCEDAL, J., "On the limited memory BFGS method for large scale optimization," Mathematical programming. Vol. **45**(1-3): pp. 503-528, 1989
[70] HUANG, Y., WANG, S., and YANG, V., "Systematic analysis of lean-premixed swirl-stabilized combustion," AIAA J. Vol. **44**(4): pp. 724-740, 2006
[71] WACKERNAGEL, H., Multivariate geostatistics: an introduction with applications. 2013: Springer Science & Business Media.
[72] JOSEPH, V.R., HUNG, Y., and SUDJIANTO, A., "Blind kriging: A new method for developing metamodels," JMD. Vol. **130**(3): pp. 031102, 2008
[73] KOEHLER, J., and OWEN, A., "9 Computer experiments," Handbook of statistics. Vol. **13**: pp. 261-308, 1996
[74] MACDONALD, B., RANJAN, P., and CHIPMAN, H., "GPfit; An R Package for Fitting a Gaussian Process Model to Deterministic Simulator Outputs," JSS. Vol. **64**(1): pp. 1-23, 2015
[75] SOBOL, I.M., "Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates," Math. Comput. Simulat. Vol. **55**(1-3): pp. 271-280, 2001
[76] WU, C.J., and HAMADA, M.S., Experiments: planning, analysis, and optimization. Vol. 552. 2011: John Wiley & Sons.
[77] SOBOL', I.y.M., "Distribution of points in a cube and integration nets," Uspekhi Matematicheskikh Nauk. Vol. **21**(5): pp. 271-272, 1966
[78] GEURTS, P., IRRTHUM, A., and WEHENKEL, L., "Supervised learning with decision tree-based methods in computational and systems biology," Molecular Biosystems. Vol. **5**(12): pp. 1593-1605, 2009
[79] FRIEDMAN, J., HASTIE, T., and TIBSHIRANI, R., The elements of statistical learning. Vol. 1. 2009: Springer series in statistics New York.
[80] BREIMAN, L., Classification and regression trees. 1st ed. 1984, New York: Routledge.
[81] CONTI, S., and O'HAGAN, A., "Bayesian emulation of complex multi-output and dynamic computer models," J. Stat. Plan. Inference. Vol. **140**(3): pp. 640-651, 2010
[82] CONTI, S., GOSLING, J.P., OAKLEY, J.E., and O'HAGAN, A., "Gaussian process emulation of dynamic computer codes," Biometrika. Vol. **96**(3): pp. 663-676, 2009
[83] LIU, F., and WEST, M., "A dynamic modelling strategy for Bayesian computer model emulation," Bayesian Anal. Vol. **4**(2): pp. 393-411, 2009
[84] HUNG, Y., JOSEPH, V.R., and MELKOTE, S.N., "Analysis of Computer Experiments With Functional Response," Technometrics. Vol. **57**(1): pp. 35-44, 2015

[85] STEIN, A., and CORSTEN, L., "Universal kriging and cokriging as a regression procedure," Biometrics. Vol.: pp. 575-587, 1991
[86] BANERJEE, S., CARLIN, B.P., and GELFAND, A.E., Hierarchical modeling and analysis for spatial data. 2014: Crc Press.
[87] MAK, S., BINGHAM, D., and LU, Y., "A regional compound Poisson process for hurricane and tropical storm damage," Journal of the Royal Statistical Society: Series C (Applied Statistics). Vol. **65**(5): pp. 677-703, 2016
[88] FRIEDMAN, J., HASTIE, T., and TIBSHIRANI, R., "Sparse inverse covariance estimation with the graphical lasso," Biostatistics. Vol. **9**(3): pp. 432-441, 2008
[89] BIEN, J., and TIBSHIRANI, R.J., "Sparse estimation of a covariance matrix," Biometrika. Vol. **98**(4): pp. 807-820, 2011
[90] QIAN, P.Z.G., WU, H., and WU, C.F.J., "Gaussian Process Models for Computer Experiments With Qualitative and Quantitative Factors," Technometrics. Vol. **50**(3): pp. 383-396, 2008
[91] NOCEDAL, J., and WRIGHT, S.J., Numerical optimization 2nd. 2006, Springer.
[92] BAZAROV, V., YANG, V., and PURI, P., "Design and dynamics of jet and swirl injectors," Liquid Rocket Thrust Chambers: Aspects of Modeling, Analysis, and Design. Vol. **200**: pp. 19-103, 2004
[93] IMHOF, J.-P., "Computing the distribution of quadratic forms in normal variables," Biometrika. Vol. **48**(3/4): pp. 419-426, 1961
[94] DAVIES, R.B., "Numerical inversion of a characteristic function," Biometrika. Vol. **60**(2): pp. 415-417, 1973
[95] DAVIES, R.B., "Algorithm AS 155: The distribution of a linear combination of χ 2 random variables," Journal of the Royal Statistical Society. Series C (Applied Statistics). Vol. **29**(3): pp. 323-333, 1980
[96] CASTA**14**(2): pp. 397-415, 2005
[97] LIU, H., TANG, Y., and ZHANG, H.H., "A new chi-square approximation to the distribution of non-negative definite quadratic forms in non-central normal variables," Computational Statistics & Data Analysis. Vol. **53**(4): pp. 853-856, 2009
[98] TEAM, R.C., "R: A language and environment for statistical computing." Vol., 2013
[99] DUCHESNE, P., and DE MICHEAUX, P.L., "Computing the distribution of quadratic forms: Further comparisons between the Liu–Tang–Zhang approximation and exact methods," Computational Statistics & Data Analysis. Vol. **54**(4): pp. 858-862, 2010
[100] GOOVAERTS, P., Geostatistics for natural resources evaluation. 1997, New York: Oxford University Press.